VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

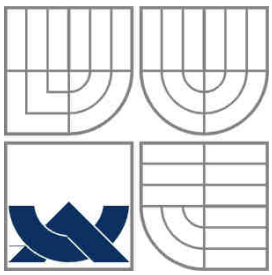
DOKUMENTACE PROCESŮ

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

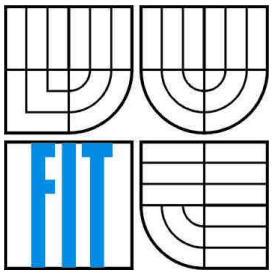
AUTOR PRÁCE
AUTHOR

MARTIN DOSTALÍK

BRNO 2007



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

DOKUMENTACE PROCESŮ
PROCESS DOCUMENTATION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

MARTIN DOSTALÍK

VEDOUCÍ PRÁCE
SUPERVISOR

RNDr. JITKA KRESLÍKOVÁ, CSc.

BRNO 2007

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2006/2007

Zadání diplomové práce

Řešitel: **Dostalík Martin**
Obor: Výpočetní technika a informatika
Téma: **Dokumentace procesů**
Kategorie: Softwarové inženýrství

Pokyny:

1. Seznamte se s metodikou IBM Rational Unified Process (RUP).
2. Seznamte se s principy managementu procesů.
3. Specifikujte požadavky na systém pro dokumentaci procesů.
4. Navrhněte programový systém pro dokumentaci procesů.
5. Zvolte vhodné implementační prostředí a realizujte prototyp navrženého systému. Funkčnost systému prověřte na vhodně zvoleném vzorku dat.
6. Zhodnoťte dosažené výsledky. Zaměřte se na použitelnost v reálném prostředí. Diskutujte možnosti dalšího rozvoje.

Literatura:

- Becker, J., Kugeler, M., Rosemann, M.: Process Management, Springer-Verlag, 2003, ISBN: 3-540-43499-2.
- Dyba, T., Dingsoyr, T., Moe, N. B.: Process Improvement in Practice, Kluver Academic Publishers, 2004, ISBN: 1-4020-7869-2.
- Bergström, S., Raberg, L.: Adopting the Rational Unified Process, Addison-Wesley, 2003, ISBN: 0-321-20294-5.

Při obhajobě semestrální části diplomového projektu je požadováno:

- Splnění bodů 1 - 3.

Podrobné závazné pokyny pro vypracování diplomové práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci ročníkového a semestrálního projektu (30 až 40% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním paměťovém médiu (disketa, CD-ROM), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Kreslíková Jitka, RNDr., CSc.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2006
Datum odevzdání: 22. května 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

doc. Ing. Jaroslav Zendulka, CSc.
vedoucí ústavu

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami

1. Pan

Jméno a příjmení: **Martin Dostalík**
Id studenta: 21485
Bytem: Tetčická 7, 664 47 Střelice
Narozen: 05. 01. 1982, Brno
(dále jen "autor")

a

2. Vysoké učení technické v Brně

Fakulta informačních technologií
se sídlem Božetěchova 2/1, 612 66 Brno, IČO 00216305
jejímž jménem jedná na základě písemného pověření děkanem fakulty:

.....
(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
diplomová práce

Název VŠKP: Dokumentace procesů
Vedoucí/školitel VŠKP: Kreslíková Jitka, RNDr., CSc.
Ústav: Ústav informačních systémů
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:
tištěné formě počet exemplářů: 1
elektronické formě počet exemplářů: 2 (1 ve skladu dokumentů, 1 na CD)

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizování výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti:
   - ☒ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: .....

.....	.....
Nabyvatel	Autor

Abstrakt

Cílem tohoto projektu je seznámení s řízením, analýzou a dokumentací procesů využívaných při tvorbě softwarových programů a vytvoření prototypu pro jejich podporu. Vysvětluje proč je pro firmy důležité vytvářet pracovní procesy, měřit jejich výkonnost a postupně je vylepšovat. Dále popisuje metody, které pomáhají zajistit přijetí těchto procesů v praxi a dále je řídit, zejména Rational Unified Process od firmy IBM. Také jsou zde popsány prostředky pro vývoj www aplikací a práci s nimi. Práce obsahuje specifikaci požadavků a návrh na aplikaci pro programovou podporu dokumentace procesů a popis této aplikace. Součástí diplomové práce je systém ve formě aktivních www stránek sloužící k využití knihovny procesů při vývoji softwarových projektů.

Klíčová slova

Proces, Rational Unified Process, řízení procesu, dokumentace procesu, měření procesu, softwarové inženýrství.

Abstract

The goal of this project is the identification with management, analysis and documentation of processes which are used in software development and creating prototype which support these processes. It explains why is important to create new working processes, measure their efficiency and improve them subsequently. Next, it describes the methods which help with the integration of these processes at work and, manage them, especially the Rational Unified Process from the IBM company. Then, I describe the ways how to develop www applications and, consequently, how to work with these applications. The thesis also involves the specification of the requirements and the scheme for the program support application of the process documentation and the description of this application. Another part of the thesis is the system in the active web side form serving to the utilization of a process library in the software projects developing.

Keywords

Process, Rational, Unified Process, process management, process documentation, process measurement, software engineering.

Citace

Martin Dostalík: Dokumentace procesů, diplomová práce, Brno, FIT VUT v Brně, 2007

Dokumentace procesů

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením RNDr. Jitky Kreslíkové, CSc.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Martin Dostalík
20.5.2007

Poděkování

Na tomto místě bych rád poděkoval vedoucí svého semestrálního projektu RNDr. Jitce Kreslíkové, CSc. za připomínky, cenné rady a za její vstřícný přístup.

Obsah

Úvod

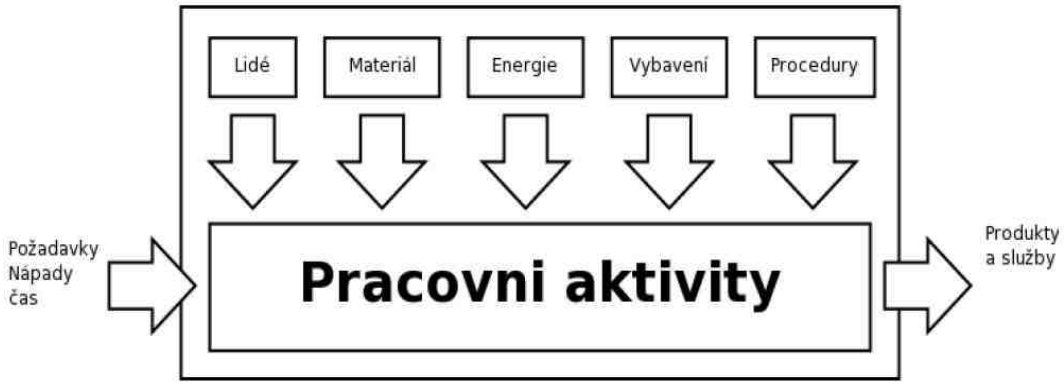
V současné době existuje obrovské množství firem, zabývajících se vývojem softwaru. Aby se v tak velké konkurenci firma prosadila, musí vyvíjet co nejkvalitnější produkty. Tyto produkty musí být nejen kvalitní, ale musí být také vyvíjeny v co nejkratším čase a s co nejmenšími náklady. Proto se vyplatí uchovávat a dále využívat znalosti získané při předchozích aktivitách, i znalosti získané ostatními. Získané znalosti uplatníme při tvorbě procesů využívaných při vytváření softwarových projektů. Tato práce se zabývá právě procesy určenými pro vývoj softwaru, jejich řízením a dokumentací a také má za úkol vytvořit prototyp systému pro podporu těchto procesů.

Diplomová práce není pokračováním ročníkového projektu. Vychází ze semestrálního projektu, ve kterém byla popsána teorie využití procesů při vývoji softwaru a navazuje na něj systémem programové podpory. V kapitole 3 semestrálního projektu je jednoduchá specifikace požadavků, jenž je v kapitole 5.1 diplomové práce upřesněna a doplněna.

1 Procesy a jejich řízení

1.1 Definice procesu

Proces můžeme definovat jako „soubor vzájemně souvisejících nebo vzájemně působících činností, které přeměňují vstupy na výstupy“ (podle ČSN ISO 9000, Systémy managementu jakosti – Základy, zásady a slovník). Pro oblast vývoje software lze vývojový proces popsat jako „sadu činností, metod, praktik a transformací, které se používají k vytvoření a k údržbě softwarových produktů“. Znázornění procesu je na obrázku obr. 1.



Obr. 1 – Definice procesu

Konkrétněji lze tento proces definován jako:

- činnosti jež mají být provedeny
- role a odpovědnosti všech, kteří se na činnostech podílejí
- metriky využívané pro plánování, odhadování, sledování a zdokonalování procesů
- produkty, jež mají být vytvořeny
- návaznosti jednotlivých činností v procesu
- techniky a nástroje používané k provádění činností

1.2 Dokumentace procesů

Procesy jsou dokumentovány a analyzovány na základě dokumentovaných modelů procesů. Procesy a jejich modely lze popsat různými způsoby. Pro zajištění správného pochopení popsaného procesu je vhodné použít formální notaci. Těchto notací existuje velké množství a my musíme vybrat

tu, která nám nejlépe vyhovuje. Pokud vybereme špatně, může se stát že nevyužijeme všech výhod dokumentovaného procesu.

Notace lze rozdělit do čtyř skupin:

1.2.1 Strukturované grafické pohledy

Pohled je popsán pomocí grafů a obrázků, což velmi napomáhá pochopení procesu zejména u osob, které nemají s procesy zkušenosti. Používají se např. vývojové diagramy – flowcharts, kterými lze zobrazit pracovní proces. Tyto diagramy mohou být doplněny textovými poznámkami.

1.2.2 Programové pohledy

Vycházejí z toho, že na proces lze pohlížet jako na program, který má být proveden. Proces je popsán jako program v jazyku podobném programovacím jazykům. Výhodou těchto pohledů je snadné počítačové zpracování, ale horší pochopitelnost.

1.2.3 Pohledy na pohyby v systému

Vycházejí z principů a technik pro vytvoření pohledů na proces jako dynamický systém. Jejich výhodou je možnost velmi přesného popisu procesů v systému a v budoucnu možnost snadné analýzy procesů, ale jejich pochopitelnost je kvůli rozsáhlosti popisu menší.

1.2.4 Pohledy pomocí Petriho sítí

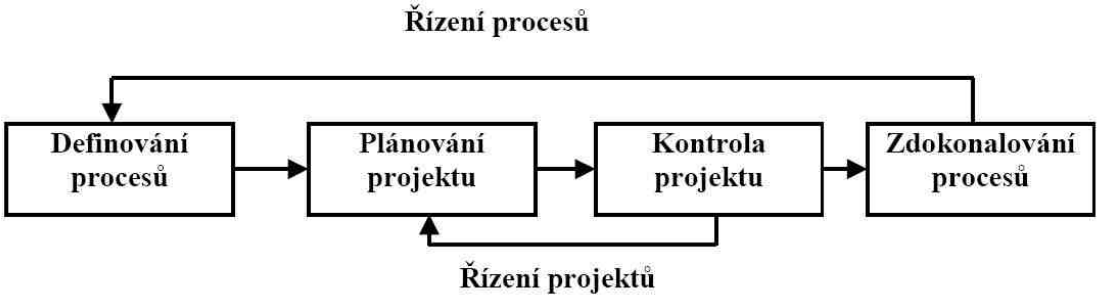
Používají matematicky založené grafické notace pro reprezentaci dynamických a distribuovaných procesů. Umožňují analýzu procesů vyspělými nástroji, jejich pochopitelnost je různá.

V rámci uchování širšího pohledu na proces používáme pojem softwarový proces nejen na všeobjímající proces celé organizace, ale na všechny procesy i dílčí podprocesy. To zahrnuje i plánování, návrh, testování a spravování stejně jako jakékoliv dílčí úlohy a aktivity. Z definice projektu vyplývá, že proces a jeho provádění je v drtivé většině případů ovlivněn omezeními. Proces musí být proveden v určitém omezeném čase, náklady se musí vejít do stanoveného rozpočtu a musíme vystačit s přidělenými prostředky. Prostředky (zdroje) mohou být technického charakteru jako hardwarové vybavení firmy (počet počítačů, počet licencí používaného softwaru), dále např. finanční nebo časové a také nesmíme zapomenout na lidské zdroje (odborníci, kteří jsou k dispozici v čase vývoje projektu). Kdybychom tato omezení neměli, téměř bychom nepotřebovali uplatňovat řízení procesů. Ve reálném světě bohužel tyto omezení jsou, proto musíme řízení do vývoje zapojit.

Podrobnější informace o této problematice lze nalézt např. v [6].

1.3 Řízení procesů

Procesní řízení (process management) je soubor metod a nástrojů používaných k získávání, definování, distribuci, provádění a zdokonalování nejlepších praktik využitelných při zajišťování procesů. Filosofie nejlepších praktik vyžaduje transformaci vědomostí o činnostech v organizaci do podoby procesů, které mohou být využívány kdykoliv v případě potřeby, kteroukoliv osobou která na to má oprávnění. Tak dochází k vytvoření prostředí, ve kterém každý softwarový projekt těží ze zkušenosti dřívějších projektů a neustále dochází ke zdokonalování procesů vývoje software. Viz obr. 2.



Obr. 2 – Řízení procesů

Řízení rozvoje firemních procesů zahrnuje:

- získání či vytvoření vlastní knihovny (knihoven) s nejlepšími oborovými praktikami
- její rozšíření o nejlepší firemní praktiky
- vytvoření prostředí a situace, kdy vedoucí projektů využívají nejlepších praktik k definování a zajišťování projektů
- zajištění toho, že členové vývojového týmu realizují projekt s využitím nejlepších praktik
- zdokonalování nejlepších praktik na základě měřitelných zkušeností s jejich aplikováním a zpřístupnění zdokonalených procesů pro další využití
- využívání technologie pracovních toků pro podporu těchto procesů

Zpočátku je možné s procesy pracovat i bez počítačové podpory. V tomto případě jsou procesy pouze zdokumentovány a opakovaně používány. Zlepšování procesů je prováděno náhodně a úspěšnost jednotlivých projektů může kolísat. Jakmile začneme klást důkaz na optimalizaci procesů, je použití informační a komunikační technologie nezbytné.

Softwarové nástroje procesního řízení by měly mj. umožňovat:

- komunikaci členů vývojového týmu
- snadnou dostupnost knihovny procesů

- pro oprávněné osoby možnost definovat a zdokonalovat obsah knihovny a modely metrik pro měření efektivity procesů
- pro manažery procesů možnost zvolit nejvhodnější proces, naplánovat projekt, rozdělit práci a sledovat její aktuální plnění
- přístup členům týmu k definicím přidělených činností projektu a usnadnit jejich provádění

Nevýhodami procesního řízení je počáteční investice do zdokumentování procesů, získání zkušeností pracovníků s chápáním definovaných procesů a také náklady na udržování procesů a jejich optimalizaci. O řízení procesů pojednává publikace [1].

1.3.1 Role a odpovědnosti

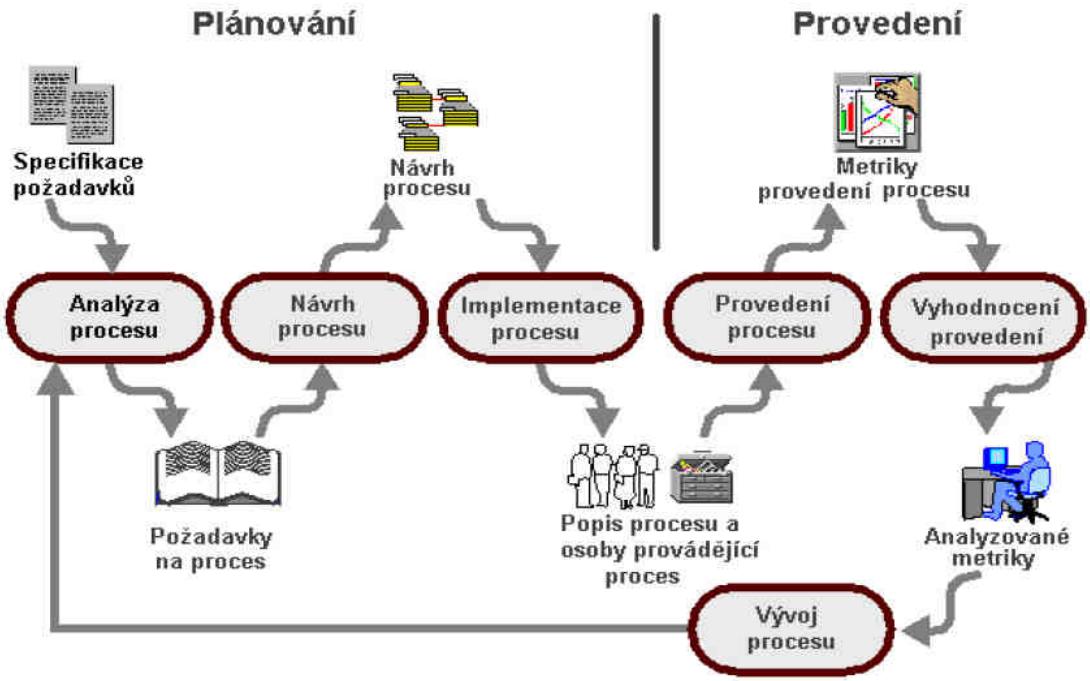
Při procesním řízení dělíme role a odpovědnosti angažovaných zaměstnanců do tří hlavních oblastí:

Podnikový metodik neboli manažer procesů definuje a řídí obsah knihovny nejlepších praktik a realizuje potřebná zdokonalení procesů. Tuto roli většinou zastává více osob.

Manažer projektu vytváří strukturu projektu z nejvhodnějších procesů a po přizpůsobení specifikům nového projektu vytvoří plán projektu. Dále sleduje postup práce, produkty a metriky.

Členové týmu provádějí proces tak, že vykonávají přidělenou práci, vytváří požadované produkty a informují manažera projektu o postupu projektu.

1.3.2 Životní cyklus řízení rozvoje procesu



Obr. 3 – Životní cyklus řízení rozvoje procesu

Cyklus řízení prací na rozvoji procesu začíná definicí procesu. Ta se skládá z analýzy, návrhu a implementace. Dále do tohoto cyklu patří provedení a vyhodnocení procesu a jeho vývoj. Životní cyklus řízení rozvoje procesu je na obrázku obr. 3 (převzato z [5]).

1.3.2.1 Analýza procesu

Analýza procesu znamená shromáždění požadavků na proces, tvorbu analytických modelů, určení účelu popisu procesu a zhodnocení požadavků. Základními požadavky jsou vstupy, výstupy, metriky, techniky, nástroje a normy, které musí proces splňovat. Všechny tyto požadavky by měly být zdokumentovány a zaneseny v modelech pomáhajících pochopit všechny aspekty daného problému. Proces by měl být popsán tak, aby byl srozumitelný z různých pohledů, protože jinak se na proces dívá manažer procesu a jinak členové týmu. Je také možné najít již hotový podobný proces v knihovně modelů a inspirovat se jím. Výsledek analýzy bývá konzultován se zákazníkem.

1.3.2.2 Návrh procesu

Návrh procesu zahrnuje modelování procesu a vyhodnocení vytvořeného modelu. Model je hodnocen procesními inženýry, osobami, které budou proces provádět a vedením organizace. Při modelování procesu by měli být zachyceny všechny detaily týkající se procesu. Ty lze získat například rozhovory s lidmi, které mají s daným procesem zkušenost v praxi nebo analýzou průběhu dřívějších modelů. Také se v této fázi určují metriky, které budou v průběhu projektu ohodnocovány a analyzovány. Modely procesů se využívají i k dokumentaci a následnému zhodnocení stávajících procesů.

1.3.2.3 Implementace

Při implementaci se vytváří průvodci procesem, které by měli být srozumitelné i osobám málo zkušeným v používání procesů. Také se mohou vytvářet a instalovat skripty pro provádění procesů. Probíhají školení osob, které budou proces provádět. Při zahájení konkrétního projektu se většinou proces, ze kterého vychází, upraví tak, aby vyhovoval specifikům nového projektu.

1.3.2.4 Provedení procesu

Provedení procesu (resp. projektu podle určitého procesu) je vykonávání definovaných postupů. Jednotliví členové týmu plní naplánované úkoly, manažer řídí projekt. Některé skripty mohou být zpracovány strojově. Také by měly být zaznamenávány metriky určené při návrhu procesu (např. náklady, doba provedení, kvalita…).

1.3.2.5 Vyhodnocení provedení

V této fázi dochází k vyhodnocení metrik získaných při provádění procesu. Podle výsledků určíme úspěšnost projektu a odchylky od původních požadavků. V případě úspěšného posouzení projektu můžeme nové postupy, jenž se osvědčily, převzít do procesů ze kterých budou vycházet podobné projekty. V případě neúspěchu projektu je nutné určit klíčové body, proč byl projekt neúspěšný a zdokumentovat je, aby nemohlo dojít k jejich opakování. Podněty na zlepšení nemusí být formulovány pouze ze získaných metrik, ale také od lidí, kteří na projektu pracovali a získali nové poznatky.

1.3.2.6 Vývoj procesu

Proces může být na základě poznatků z realizovaných projektů průběžně zlepšován a zdokonalován pro úspěšnější a efektivnější použití v dalších projektech. Analýzo a zlepšováním procesů se zabývá publikace [2].

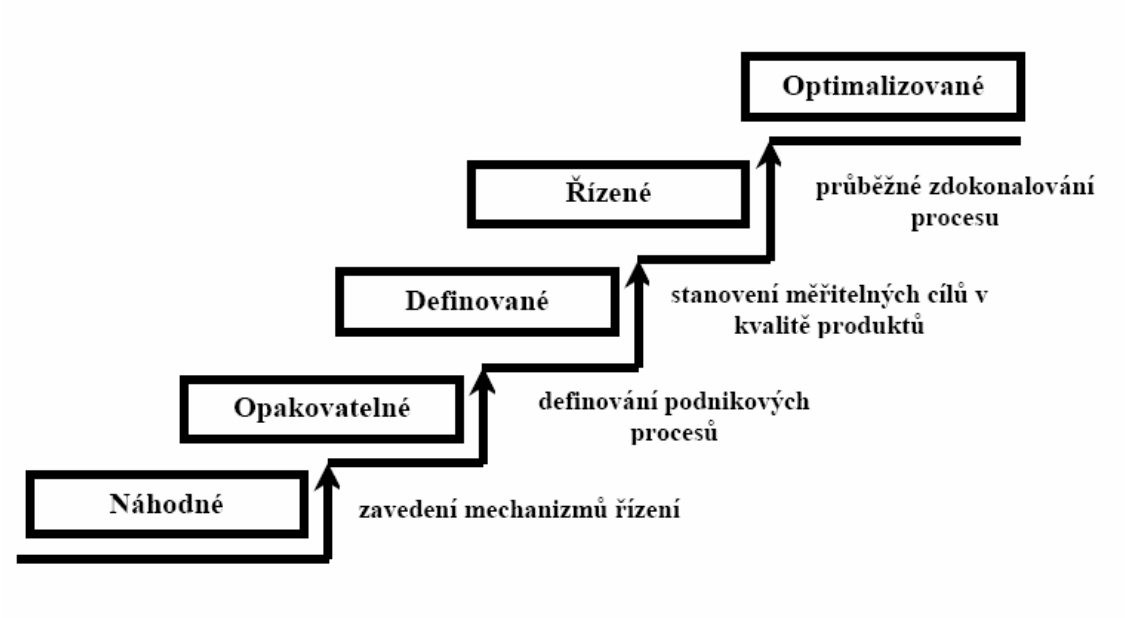
Tento cyklus řízení procesu neprobíhá jen jednou, ale je vykonáván opakovaně, většinou při realizaci konkrétních projektů odvozených z procesu.

1.3.3 Klasifikace organizací podle procesů

K rozdělení organizací podle vyspělosti procesů se používá klasifikace CMM – Capability Maturity Model, což znamená model pro určení (a porovnání) schopnosti a vyspělosti organizace se zaměřením na úroveň využívání procesů. Model SW-CMM je určen organizacím vyvíjejícím software jako pomoc při zlepšování řízení procesů probíhajících v organizacích, tím, že poskytuje klasifikaci, podle které se může organizace zařadit a dále usilovat o vyšší stupeň plněním požadavků tohoto stupně.

Klasifikace je vyvíjena Institutem Softwarového Inženýrství (Software Engineering Institute – SEI) od roku 1986. V současnosti se SEI zaměřuje na modely pro začlenění (integraci) CMM do organizací – CMMI (CMM Integration), které vychází z původních modelů CMMI-SW (vývoj software), CMMI-SE (vývoj elektronických zařízení), CMMI-IPPD (model integrovaného vývoje firemních procesů), a CMMI-SS (model řízení subdodávek). Na vývoji se podílí tisíce odborníků z mnoha firem i z americké armády. Normy CMMI lze přirovnat k normě jakosti ISO 9001:2000, i když v některých parametrech se liší. V naší republice nejsou tak rozšířené, ale v západní Evropě a Americkém kontinentu některé firmy považují třetí úroveň CMMI za prestižnější než certifikaci podle ISO 9001:2001. Tato tématika je více popsána v [9].

V klasickém CMM je definováno 5 úrovní organizací, tzv. Maturity Levels (tyto úrovně jsou stejné i v CMMI), viz obr. 4 (převzato z [5]). Pro každou úroveň jsou definovány konkrétní požadavky, které musí organizace splnit, aby mohla být považována za organizaci na dané úrovni.



Obr. 4 – Úrovně CMM

1.3.3.1 Počáteční úroveň (náhodné procesy)

Organizace na této úrovni nemají stabilní prostředí pro vývoj a získávání procesů. Pokud organizace používá řízení aspoň v některých oblastech, jeho výhody bývají znehodnocovány neefektivním plánováním a vazbami založenými na aktuálních operativních potřebách. Při krizi se ovšem upouští od plánovaných procedur a přechází se přímo k realizaci prioritních aktuálních potřeb, ale to je již situace mimo oblast procesního řízení.

Vyspělost jednotlivých procesů u organizace na této úrovni je neurčitelná, protože procesy se neustále mění i v v průběhu práce. Ani časový plán, rozpočet, funkčnost a kvalitu produktu nelze obecně předvídat. Několik stabilních základních procesů bývá k dispozici, ale jejich efektivita je určena nahodile, obvykle jen zkušenostmi a úsilím osob které je vykonávají.

1.3.3.2 Opakovatelné procesy

Organizace by již měly zavést efektivní řízení procesů pro jednotlivé projekty tak, že to umožní organizaci opakovat úspěšné praktiky, ačkoliv se jednotlivé projekty mohou lišit. Efektivní proces lze charakterizovat jako vyzkoušený v praxi, dokumentovaný. Je také zaručeno jeho používání, jeho dostatečná znalost, provádění procesu je měřeno a lze jej zlepšovat.

Organizace na této úrovni má pak zavedeny stabilní procesy pro řízení konkrétních projektů. Plánování a řízení nových projektů bývá založeno na zkušenostech získaných s podobnými projekty a tyto plány bývají díky tomu reálné a dosažitelné. Vedoucí projektů sledují náklady, časový plán a funkčnost produktu. Ale problémy s neplněním závazků projektu (termínů, nákladů) jsou řešeny, až

když se objeví. Požadavky na výsledky a na produkty, které je uspokojují, jsou zachyceny a jejich integrita je kontrolována. Existují standardy pro projekty a je ověřováno, zda jsou dodržovány.

1.3.3.3 Definované procesy

Na této úrovni již musí mít organizace standardizované procesy v rámci celé organizace. Procesy a řízení jejich vývoje jsou udržovány v kompaktním celku. Zavedené procesy pomáhají procesním a projektovým manažerům a vývojovým pracovníkům pracovat efektivněji. Organizace využívá výhodné praktiky oboru při definici svých procesů. V organizaci existuje skupina odpovědná za procesy, která procesy upravuje a hledá další možnosti jejich zlepšení.

V organizaci je zaveden program pro vzdělávání, který zajišťuje, aby měli zaměstnanci a vedení schopnosti a znalosti na provádění svěřených úkolů. Procesy a jejich skripty by měly být přístupné a čitelné, se specifikací vstupů, standardů a činností pro provedení, kontrolních mechanismů, výstupů a kritérií pro určení kompletnosti provedení procesu.

1.3.3.4 Řízené procesy

Organizace musí mít definovány měřitelné požadavky na kvalitu jednotlivých produktů a procesů. Produktivita a kvalita je měřena a poměřována pro všechny důležité činnosti procesů mezi všemi projekty v organizaci.

Používá se databáze procesů mj. pro sbírání a analýzu dat z definovaných procesů. Všechny procesy jsou dokumentovány s dobře definovanými a konzistentními metrikami, které umožňují procesy a produkty hodnotit. Kvalitu výsledných produktů lze předvídat a její úroveň bývá vysoká. Rizika plynoucí z určení nových - třeba i problémových - aplikací jsou známa a důsledně řešena.

1.3.3.5 Optimalizované procesy

Organizace na této úrovni se celé zaměřují na průběžné zlepšování procesů. Mají k dispozici prostředky, jak odhalit slabá místa procesů. Mají vypracovaný způsob jejich posílení s cílem zabránit případným budoucím problémům a kolizím.

Pro procesy existují statistiky jejich úspěšnosti a ty jsou využívány při analýze nákladů a zisků nových technologií a pro případnou následnou změnu procesů pro jejich lepší využití. Jsou určeny možnosti použití nejlepších praktik pro daný obor. Organizace systematicky analyzují chyby pro určení jejich původu, chyby jsou zdrojem poučení. Procesy jsou analyzovány pro prevenci opakování poznaných problémů.

1.3.4 Měření procesů

Jednou z důležitých činností pří řízení procesů je jejich měření. Měření výkonnosti může probíhat dvěma způsoby. Buď měřením vlastností produktu, který proces produkuje, nebo měřením vlastností samotného procesu. Při splnění podmínek měření a řízení můžeme pomocí informací z minulosti určit jak se proces chová nyní a jak se pravděpodobně bude chovat v budoucnu.

Měřit výkonnost procesů můžeme také měřením vlastností zdrojů nebo prostředí, které proces podporuje. Klíčem k měření výkonnosti je volba vhodných atributů procesu. Měříme nejen atributy odrážející zamýšlený účel procesu, ale bereme v úvahu i místa, kde můžeme očekávat problémy. Na obrázku obr. 5 (zdroj - [4]) vidíme některé měřitelné vlastnosti objektů v softwarovém procesu.

Měřitelné vlastnosti objektů v softwarovém procesu			
Věci přijaté, či užité	**Aktivity a jejich části**	**Věci spotřebované**	**Věci produkované**
Změny ◆ Typ ◆ Datum ◆ Velikost ◆ # obdrženo Požadavky ◆ Požadavky na stabilitu ◆ # identifikováno ◆ % v návrhu ◆ % v programování Zprávy o problému ◆ typ ◆ datum ◆ velikost ◆ původ ◆ striktnost ◆ # obdrženo Fondy ◆ peníze ◆ rozpočet ◆ stav Lidé ◆ roky zkušeností ◆ typ vzdělání ◆ % trénováno v XYZ Zařízení a prostředí ◆ stop čtverečních na zaměstnance ◆ hluk ◆ osvětlení ◆ # personálu v kanceláři ◆ investice do nábytku pro zaměstnance ◆ hodin používání počítače ◆ % využité kapacity	Cesty ◆ čas zpracování ◆ výkon ◆ rozptýlení ◆ zpoždění ◆ rezervy Délka, velikost ◆ Fronty ◆ Zásobníky ◆ Vyrovnávací paměti	Úsilí ◆ # hodin vývoje ◆ # hodin přepracování ◆ # hodin příprav ◆ # hodin schůzek Čas ◆ Výchozí čas/datum ◆ Konečný čas/datum ◆ Délka procesu/úlohy ◆ Čekací doba Peníze ◆ Cena k datu ◆ Cena povolení ◆ Cena přepracování	Stav pracujících jednotek ◆ # vyrobeno ◆ # naprogramováno ◆ # otestováno Velikost pracujících jednotek ◆ # požadavků ◆ # funkčních bodů ◆ # řádků kódu ◆ # modulů ◆ # objektů ◆ # bytů v databázi Výstupní množství ◆ # akčních položek ◆ # schválení ◆ # nalezených chyb Výsledky testu ◆ # úspěšných případů testů ◆ % pokrytí testy Architektura programu ◆ fan-in ◆ fan-out Změny ◆ typ ◆ datum ◆ velikost ◆ vydané úsilí Problémy a defekty ◆ # zpráv ◆ hustota chyb ◆ typ ◆ původ ◆ rozšíření podle typu ◆ rozšíření podle původu ◆ # otevřených ◆ # uzavřených Kritické využití zdrojů ◆ % využité paměti ◆ % využité kapacity cpu ◆ % využité kapacity I/O

Obr. 5 – Příklad měřitelných objektů v softwarovém procesu

Důležité je přesně definovat co se bude měřit a jak, aby měření byla nezávislá na osobách které jej provádějí a výsledky byly jednoznačné. Pomocí získaných informací zjistíme, jestli je proces stabilní a snažíme se ho kontrolovat. To znamená držet ho v hranicích jeho optimálního výkonu. Jakmile je proces pod kontrolou, je nutné ho v tomto stavu udržovat, protože pokud se vymkne kontrole, je velmi nákladné dostat ho zpět. I když je proces definovaný a pod kontrolou, nemusí ještě být schopen splnit požadavky zákazníka nebo organizace. V takovém případě je nutno proces vylepšit.

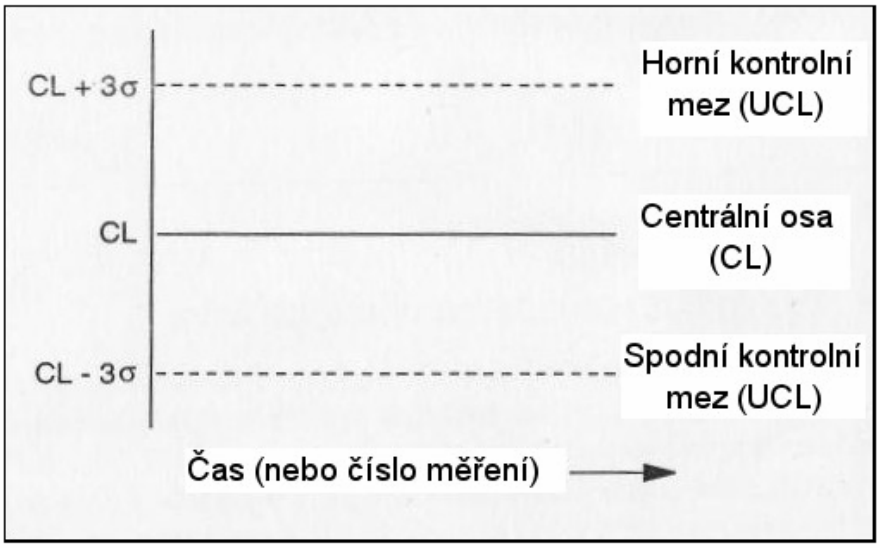
1.3.4.1 Měření stability procesu

Pro testování stability procesu potřebujeme vědět jak variabilní jsou hodnoty uvnitř podskupiny naměřených hodnot v každém měřícím bodu a potřebujeme srovnávat tyto změny mezi jednotlivými podskupinami. Přesněji řečeno potřebujeme určit, zda se změny v průběhu času shodují se změnami, které nastávají uvnitř podskupiny. Také potřebujeme zjistit možné odchylky nebo posuny od hlavního směru měřených údajů. Jednou z používaných technik pro ustanovení provozních mezí pro přijatelnou změnu je statistické řízení procesů (SPC – Statistical Process Controll). SPC využívá ke znázornění hodnot a mezí kontrolní diagramy. Struktura těchto diagramů je znázorněna na obrázku obr.6 (zdroj – [4]).

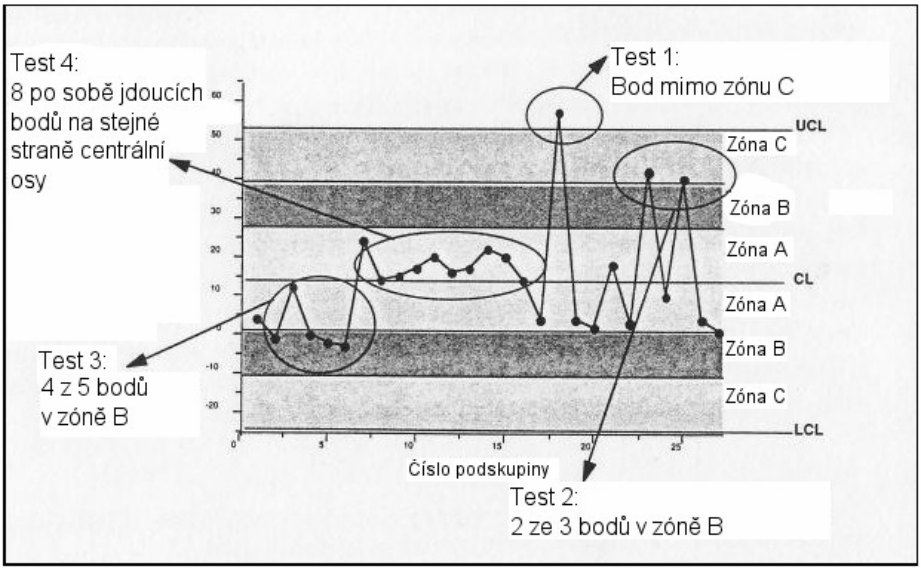
Každý kontrolní diagram má centrální osu (CL) a limity na obou stranách centrální osy. Obojí reprezentuje odhady vypočtené z předchozího průběhu procesu. Jejich hodnoty nemohou být určeny předem, záleží na aktuálním stavu procesu. Hodnota CL je obvykle určena průměrem naměřených hodnot. Kontrolní limity bývají nejčastěji určeny možným rozptylem od centrální osy ±3 sigma, kde sigma je standardní statistická odchylka. Ke zjištění nestability se používají čtyři testy, viz obrázek obr. 7 (zdroj – [4]):

- Test 1: Jedna hodnota je mimo kontrolní meze 3-sigma
- Test 2: Alespoň dvě ze tří po sobě jdoucích hodnot jsou za hranicí 2-sigma a leží na stejné straně centrální osy.
- Test 3: Alespoň čtyři z pěti po sobě jdoucích hodnot jsou za hranicí 1-sigma a leží na stejné straně centrální osy.
- Test 4: Alespoň osm po sobě jdoucích hodnot leží na stejné straně centrální osy.

Testy 1, 2 a 3 jsou nazývány testy běhu (run tests) a jsou založeny na předpokladu že data po sobě jdoucí budou mít mezi sebou nějakou statistickou závislost. Pokud používáme všechny čtyři testy, nestabilita procesu se projevuje tím, že nastane jedna z daných podmínek. Používání všech testů místo jednoho snižuje počet falešných poplachů.



Obr. 6 – Základní struktura kontrolního diagramu



Obr. 7 – Čtyři efektivní testy na stabilitu procesu

1.3.4.2 Kontrolní grafy

Ke znázornění průběhu a analýze chování procesů se používají různé kontrolní grafy, např. grafy průměrů (X-bar) a rozmezí (R), grafy individuální a pohyblivých rozmezí (XmR) pro diskrétní nebo spojitá data, c grafy, u grafy nebo z grafy.

1.3.4.2.1 X-bar a R grafy

Grafy pro průměr (X-bar grafy) a vzdálenost (R) se používají k zobrazení chování procesů při možnosti mnoha měření v krátkém časovém intervalu za stejných podmínek, měření procesu jsou sloučena do skupin.X-bar grafy vypovídají o hlavní tendenci procesu a o změnách mezi skupinami v průběhu času, odpovídající R grafy indikují změny ve skupinách. Použitím pravidel detekce

stability z kapitoly 2.3.4.1 můžeme určit jestli je proces stabilní a zjistit body ve kterých došlo k ovlivnění průměrů nebo vzdáleností skupin a proces se dostal mimo kontrolu.

1.3.4.2.2 Grafy individuálních a pohyblivých rozmezí (XMR) pro spojitá data

Pokud jsou měření široce rozprostřena v čase nebo pokud je každé měření použito samo o sobě k vyhodnocení nebo kontrole procesu, používá se spíše než graf jednotlivých hodnot než průměrů. Princip těchto grafů je ten, že pokud podskupiny mohou obsahovat nenáhodné komponenty, snížíme vliv nenáhodných jevů zmenšením podskupin na co nejmenší velikost. Nejmenší možná velikost podskupiny je 1. Pomocí XMR grafů detekujeme snadněji než X-bar grafy:

- cykly (regulární opakování vzorců)
- trendy (spojité pohyby nahoru nebo dolů)
- mixy (přítomnost více než jedné distribuce)
- seskupování a svazky (seskupení měření v určitých bodech)
- vztahy mezi obecnými vzory seskupení a specifikací

1.3.4.2.3 C grafy a u grafy

C grafy a u grafy mají v oblasti vývoje softwaru nejširší využití. Příkladem vhodného použití je měření počtu chyb objevených během testování nebo kontroly kódu. Použití c grafu nebo u grafu závisí na oblasti výskytu jevu. Pro měření počtu chyb na tisíc řádků kódu se použije u graf, pro použití c grafu jsou vhodné situace jako měření počtu chyb na jeden modul nebo test.

1.3.4.2.4 Z grafy

Při vytváření u grafů se používají proměnlivé kontrolní limity, které mohou způsobit, že graf je těžko analyzovatelný. Abychom se těmto problémům vyhnuli, používáme z grafy. Jednotlivé hodnoty i průměrná hodnota se přepočítají na poměry v jednotkách sigma. Nalezení nenáhodných vzorů chování je tak snazší.

O této problematice se lze dozvědět více v [4].

2 The Rational Unified Process

Každá organizace podílející se na vývoji softwaru (a nejen softwaru), má za cíl vyvíjet kvalitní produkty. Důležité je vytvářet tyto produkty rychle, plánovaně a hlavně efektivně. Vývoj softwaru je však velmi složitý, tím pádem obecný popis jak ho vyvinout je také velmi složitý. Pomoci může přijetí a zavedení RUP.

2.1 RUP produkt

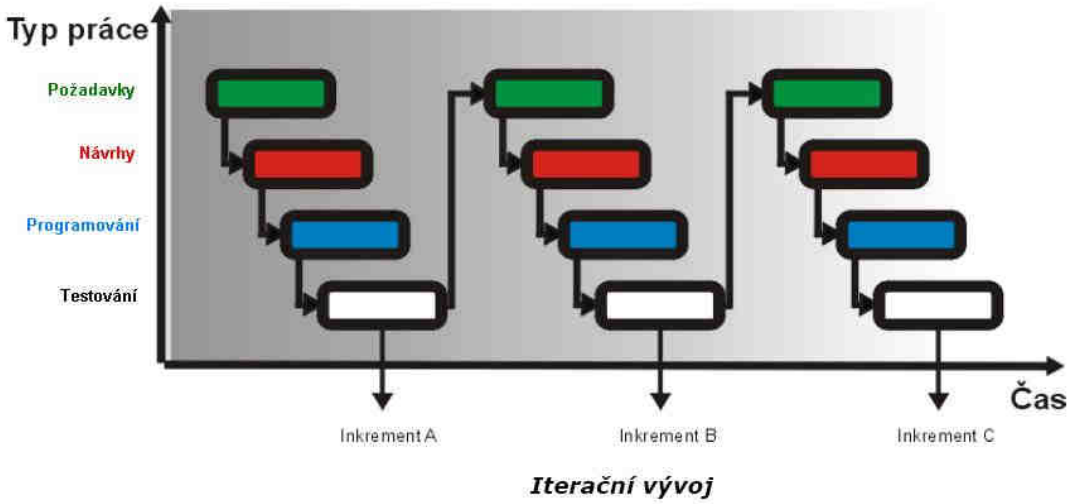
RUP je „balíček“ obsahující zkušenosti a znalosti zkušených manažerů softwarových projektů, profesionálních testerů, softwarových architektů, softwarových vývojářů a ostatních odborníků pracujících v softwarovém inženýrství. Tyto znalosti byly shromážděny a použity pro popis různých odvětví vývoje softwaru. RUP je více než jen „způsob jak myslet“ nebo metoda prezentovaná v knížce. Je to framework, rámec, pro proces vývoje softwaru. Firma si z tohoto rámce vybere části hodící se pro její proces.

Hlavní zaměření produktu je jak vyvíjet software. Rady by měly být aplikovány na mnoho lidí pracujících v různých společnostech a organizacích s různými zaměřeními. To znamená že popisy nejsou příliš specifické. Všechno speciální a více či méně unikátní pro určitou společnost nebo projekt bylo odstraněno. Proto Vám RUP neporadí věci jako: „Pokud očekáváte 300 uživatelů, kteří si současně objednávají knihu pomocí webové aplikace, měli byste ve vašem návrhu použít strukturu X a produkt Y pro komunikaci s databází“. Místo toho Vám řekne: „Uvažujte, jestli Vašim potřebám lépe vyhovuje přístup „tenký klient“ (thin-klient) nebo „silný klient“ (thick klient). Pro každé téma poskytuje RUP rady o potřebných krocích a rozhodnutích, která musí být provedena. Připomene Vám: „Mysleli jste na toto a toto?“ a „Pokud jste v této situaci, můžete vybírat mezi těmito možnostmi.“ RUP umožňuje společnostem zaměřit se raději na vytváření systému než na vytváření a údržbu procesu pro vytváření systémů.

2.2 Struktura RUP

Struktura RUP odráží jak je aktuální projekt plánován a organizován. Dříve probíhal vývoj a plánování podle tzv. vodopádového vývoje. Celý projekt byl naplánovaný najednou a všechny

požadavky byly definovány na začátku. Navrhování a psaní kódu bylo situováno do středu a testování se provádělo až na konci před předáním. Tento přístup přináší mnoho problémů, například nemožnost změny nebo upravení požadavků během projektu (je pouze jedno plánovací zasedání na začátku) nebo pozdní objevení chyb v důsledku testování pouze na konci. Další problém je, že u rozsáhlých projektů není prakticky možné naplánovat a specifikovat vše na začátku. Proto RUP používá iterační přístup, viz obr. 8 (převzato z [3]).



Obr. 8 – Iterační vývoj

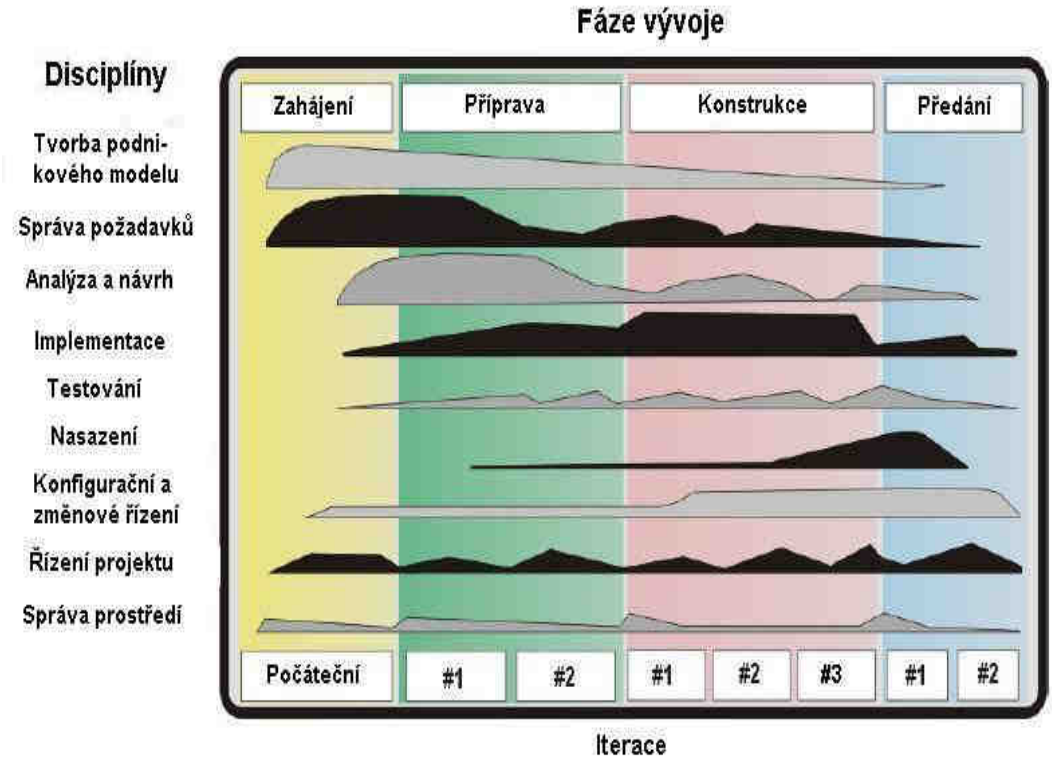
Iterační přístup spočívá v rozdělení celého projektu na čtyři fáze, přičemž každá z nich může sestávat z několika (nejméně jedné) iterací. V každé iteraci probíhá vodopádový vývoj. To umožňuje celý projekt lépe řídit. Po každé iteraci přesně víme v jakém je projekt stádiu, můžeme postupně měnit a upravovat požadavky a díky průběžnému testování snadněji odhalíme a opravíme případné chyby. Také celkem brzy získáme spustitelnou verzi.

Plánování v RUP je řízeno riziky. Nejčastější rizika jsou náklady, čas a požadavky. Tyto rizika spolu vzájemně souvisí. Když na začátku špatně odhadneme čas na vytvoření projektu, na konci můžeme zjistit že nestihneme splnit všechny požadavky nebo budeme nuceni pro jejich splnění zvýšit náklady. Na začátku projektu je nutné sestavit seznam rizik a určit postup pro jejich zmírnění a eliminaci. Tento seznam je nutné udržovat a doplňovat během celého projektu. Získané znalosti o rizicích pak přeneseme do plánu projektu, což nám přinese určité výhody. Můžeme nejdříve začíst s částí projektu které se nejvíc obáváme a u které hrozí největší riziko. Pokud zjistíme že např. technický návrh této problémové části je špatný, je možné jej změnit aniž bychom ztratili mnoho odvedené práce a tím i prostředků vynaložených na tuto práci. Tím že včas identifikujeme rizika se zlepšuje předvídatelnost projektového plánu i rozpočtu a s postupem prací se ještě zvyšuje.

Dalším důležitým prvkem je architektura. Hlavním cílem vývojové fáze Příprava (viz níže) je vytvoření architektury systému. Vytvoření znamená sestavení modelu včetně podsystémů a jejich vzájemných vztahů (rozhraní). Architektura RUP musí být spustitelná (musí být napsán kód a vykonány testy). Před dokončením projektu je pak architektura verifikována implementací a testováním. RUP využívá komponentové architektury. Dříve vytvořená komponenta může být použita znova, což vede k úspoře zdrojů.

Plánování v RUP je také řízeno případy užití, tzv. Use Case. Pomocí Use Case se obvykle vyjadřují funkční požadavky. Použitím Use Case lze snadno zjistit jestli byli stanoveny testy užití pro všechny určené případy užití systému. Také to pomáhá při sjednocení pohledu zadavatele a vývojového týmu na vyvíjený systém. Use Case diagramy lze modelovat pomocí univerzálního modelovacího jazyka (UML).

Životní cyklus projektu je podrobněji znázorněn následujícím obrázkem (obr. 9 - převzato z [3]).



Obr. 9 – Životní cyklus RUP

2.2.1 Disciplíny spojené s vývojem projektu v RUP

Různé oblasti vývoje softwaru byly v RUP zahrnuty do devíti disciplín. Pomocí dokumentace pro každou disciplínu lidé naleznou jednu nebo více rolí, které by mohli během projektu zastávat. Každé roli přísluší činnost (nebo více činností), které může člověk provádět. Tyto činnosti jsou v RUP

důkladně popsány. Během činností vznikají produkty – artefakty, jako dokumenty modely nebo kusy kódu.osoba zastávající příslušnou roli je za tyto artefakty zodpovědná. Při tvorbě malého projektu může jeden člověk přijmout více rolí, naopak při u velkých projektů se stává že zodpovědnost za jednu roli má více zaměstnanců (například tým architektů může hrát roli softwarového architekta).

Disciplíny RUP jsou:

- Tvorba podnikového modelu (Bussines Modeling)
- Správa požadavků (Requirements)
- Analýza a návrh (Analysis & Design)
- Implementace (Implementation)
- Testování (Test)
- Nasazení (Deployment)
- Konfigurační a změnové řízení (Configuration & Change Management)
- Řízení projektu (Project Management)
- Správa prostředí (Environment)

2.2.2 Fáze vývoje projektu RUP

Dokumentace získaná pomocí disciplín Vám neřekne moc o časovém rozdělení vývoje. Neříká nic o tom kdy začít činnost, ani o tom, v jakém časovém období musí být dokončen určitý artefakt (meziprodukt) před jeho předáním dalšímu členovi týmu jiné disciplíny. Proto jsou potřeba fáze. Všechny projekty RUP jsou rozděleny do čtyř fází, z nichž každá je ukončena dosažením určitého milníku. Každá fáze může probíhat ve více iteracích.

- Zahájení (Inception) – definuje účel a rozsah projektu a jeho obchodní kontext
- Příprava (Elaboration) – analyzuje potřeby zákazníka a projektu a definuje architekturu
- Konstrukce (Construction) – vývoj designu a tvorba zdrojových kódů
- Předání (Transition) – předání artefaktu do dalšího vývojového cyklu nebo předání produktu zákazníkovi

Na obrázku obr. 9 je znázorněna míra uplatnění jednotlivých disciplín během vývoje projektu.

2.2.3 Dokumentace projektů a procesů v RUP

K dokumentaci probíhajícího procesu lze v RUP použít tyto nástroje:

- textový editor – jednoduché poznámky v textové formě.
- HTML Editor – umožňuje vytvořit webovou stránku projektu nebo organizační strukturu a také vše co lze dokumentovat textovým editorem. Umožňuje jednoduchou navigaci mezi úrovněmi organizace i mezi úrovněmi projektu.

- RUP Builder – s jeho pomocí lze vygenerovat stránku se stejnou funkčností jako webová. Při prezentaci podmnožiny RUP dovoluje „vyseknout“ části, které nejsou v dané podmnožině zahrnuty.
- MyRUP – díky funkčnosti MyRUP zahrnuté ve webových stránkách procesu generovaných RUP Buildrem může každý uživatel dokumentovat své vlastní pohledy na proces a na webovou stránku
- RUP Organizer – umožňuje propojit textový popis se základním modelem procesu.
- RUP Modeler – umožňuje vytvořit role, aktivity, artefakty a spojit je do komponentů procesu.

2.2.4 Přijetí RUP

Aby organizace mohla plně využívat přínosy RUP, musí jej organizace přijmout přímo „do srdce“ vývoje software na všech úrovních, od plánů až po realizaci. Nestačí si o tom pouze přečíst, zaměstnanci se s tímto postupem musí sžít a zvyknout si na něj. Při zavádění RUP do organizace je dobré nejdříve vytvořit pomocí RUP nějaký jednodušší pilotní projekt, který má vysokou naději na úspěch. Také je důležité správně motivovat zaměstnance, aby tento přístup neodmítali, ale naopak si ho oblíbily. Je také vhodné pozvat do organizace nějakého mentora, člověka, který má s RUP zkušenosti a poradí zaměstnancům i vedení jak nejlépe postupovat.

Více o problematice RUP lze nalézt např. v [3] nebo [8].

3 Specifikace požadavků na systém pro dokumentaci procesů

Systém pro dokumentaci procesů by měl splňovat tyto požadavky:

- Obsahovat rozsáhlou a snadno dostupnou knihovnu procesů. Tyto procesy by mělo jít hodnotit a poté upravovat a zdokonalovat podle potřeb a požadavků organizace. Také musí být možnost vytvářet nové modely procesů a pro jednotlivé projekty vybírat nejvhodnější procesy a přidělovat role a činnosti zaměstnancům (role manažera).
- Obsahovat databázi všech zaměstnanců organizace se všemi povinnými osobními údaji.
- Musí být vytvořen seznam všech projektů, u každého projektu by měly být známy údaje o jeho procesech, zdrojích (i lidských), o milnících projektu a také o jeho produktech.
- Systém by měl zahrnovat také možnost měření procesů a vyhodnocení výsledků pro následné zlepšování procesů.
- Systém může obsahovat i prostředky a metriky pro měření produktů projektu a databázi pomocí které se tyto výsledky uchovávají a vyhodnocují.
- Jednotliví zaměstnanci by měli mít definovaná přístupová práva, podle kterých mají někteří např. možnost měnit jednotlivé procesy, vkládat různé informace nebo se podílet na měření a vyhodnocování.
- Měla by být umožněna snadná komunikace členům vývojových týmů.
- Systém by měl mít jednoduché a přátelské uživatelské rozhraní, být hardwarově nenáročný, ale zároveň s rychlou odezvou na požadavky.
- Program musí umožňovat přístup zároveň více uživatelům.

4 Vývojové prostředí

Programovou podporu diplomové práce jsem vytvořil formou internetových stránek s pomocí značkovacího jazyka HTML, skriptovacího jazyka PHP a databázového systému MySQL.. V této kapitole vysvětlím proč jsem si vybral právě tyto prostředky a jak vlastně fungují.

4.1 HTML, CSS

4.1.1 HTML

HTML, Hypertext Markup Language, je značkovací jazyk pro hypertext. Je nejpoužívanějším jazykem pro vytváření internetových stránek. Jazyk HTML byl vyvinut podle standardu SGML (Standard Generalized Markup Language), který slouží k formálnímu popisu struktury dokumentů. SGML byl v roce 1986 přijat organizací ISO pod označením ISO 8879 –SGML.

První verze HTML vytvořená v roce 1991, ještě nebyla podle SGML, tomuto standardu odpovídají až verze od roku 1994. K původní specifikaci byla postupně přidána podpora grafiky, možnost interaktivního používání formulářů, dále možnost vytváření tabulek, zarovnávání textu a prvky stylu pro ovlivňování vzhledu stránek. Další verze umožňují vytváření rámců a definování stylů mimo dokument kvůli možnosti vytvoření stylů v jednom souboru a jejich použití pro více stránek zároveň. Poslední verze HTML (verze 4.0.1) byla vydána v roce 1999.

Značkovací jazyk je charakterizován množinou značek (tzv. tagy) a jejich vlastností. Tyto značky se vepisují do úhlových závorek. Většina z nich je párových, tzn. existuje úvodní a koncový tag. K úvodnímu tagu lze připsat specifikaci vlastností. Např. nějaký text označuje odstavec zarovnaný doleva. Části dokumentu mezi značkami se říká element.

Každý HTML dokument by měl mít předepsanou strukturu, některé prvky jsou nepovinné:

- Deklarace DTD (definuje typ dokumentu), verze html
- Kořenový element (mezi tagy <html></html>) reprezentuje celý document
- Hlavička dokumentu (<head></head>) obsahuje metadata dokumentu, jako název stránky, styl zobrazení nebo znakovou sadu dokumentu
- Tělo dokumentu (<body></body>)

Příklad:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>PROGRAMOVÁ PODPORA PRO DOKUMENTACI PROCESŮ</title>
<link rel="stylesheet" type="text/css" href="format.css">
<meta http-equiv="content-type" content="text/html; charset=iso-8859-2">
</head>
<body>Vlastní text dokumentu</body>
</html>
```

Při deklaraci DTD se dnes používají tyto verze:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">

- striktní, pouze strukturální značkování, neobsahuje žádné značky související s formátováním vzhledu (formátování pouze pomocí CSS, viz další kapitola)

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

- přechodová, může obsahovat všechny tagy až do verze 4.1.0 včetně - nejpoužívanější

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN">

- s podporou rámců, používá se při rozdělení okna prohlížeče pomocí rámců

Nástupcem HTML je XHTML (Extensible Hypertext Markup Language). Rozdíly oproti HTML jsou následující:

- všechny atributy musejí mít hodnoty v uvozovkách
- zákaz křížení tagů
- tagy a atributy jsou malými písmeny (XHTML je case-sensitive)
- nepárové tagy končí lomítkem (např.
)
- párové tagy jsou párové povinně
- všechny atributy musejí mít hodnotu
- interní javascript a styly se zapisují jiným způsobem
- dokument má mít XML prolog
- dokument požaduje správný doctype

XHTML používá následující verze deklarací:

XHTML 1.0 přechodové (transitional), XHTML 1.0 striktní (strict) a XHTML 1.1

Nyní se pro vývoj stránek používá HTML i XHTML.

Pro programovou část své diplomové práce jsem zvolil HTML 4.0.1 Transitional. Ke zjištění jestli jsou stránky napsány správně a neobsahují chyby lze použít některý z mnoha validátorů HTML kódu. Já jsem použil validátor CSE HTML Validator 7.01 Lite, který je volně dostupný na internetu.

4.1.2 CSS

CSS (Cascading Style Sheets, kaskádové styly) se používají od roku 1994, první verze, CSS 1, vznikla roku 1996. Vznikly jako souhrn metod pro formátování a úpravu vzhledu stránek. Používá se k úpravě obsahu HTML, XHTML a XML dokumentů. Pomocí stylů můžeme přesně určit jak bude daný element vypadat. Největší výhodou při použití CSS stylů je možnost na jednom místě definovat formát určitého elementu (např. nadpisu nebo odstavce) a tuto definici použít při každém výskytu elementu bez nutnosti vypisovat pokaždé všechny jeho atributy. Také v případě změny stačí upravit vlastnosti pouze na jednom místě a změní se nám tím vlastnosti elementů v celém dokumentu. Pokud umístíme definici stylů do samostatného souboru, lze je použít dokonce pro více dokumentů. Použitím stylů se tak kód stane kratším a přehlednějším. Pomocí stylů definujeme pravidla pro jednotlivé elementy. Každé pravidlo má dvě části, selektor (typ elementu) a definici (vlastnosti elementu). Deklarace vypadá následovně:
selektor {vlastnost:hodnota}, např. h1 {color:blue} nastaví všem nadpisům stupně 1 modrou barvu. Pokud definujeme u některého elementu stejnou vlastnost dvakrát, platí později uvedená hodnota.

Styly lze definovat třemi způsoby:

- přímo u určitého elementu
  <h1 style=”color:blue”>Nadpis</h1>
- v hlavičce dokumentu
  <style>
  h1{color:blue}
  </style>
- v externím souboru
  h1{color:blue}
  Do hlavicky HTML dokumentu musíme napsat odkaz na soubor
  <link rel=”stylesheet” type=”text/css” href=”nazevsouboru.css”>

Také lze definovat vlastní styly:

```
.nadpis {
color: #blue;
font-family: Arial, Verdana, Helvetica, sans-serif;
```

font-size: 14pt;

font-weight: bold;

}

Vlastní styl v těle dokumentu zapíšeme jako Nadpis

V implementaci jsem použil vlastní styly v kombinaci s definicí vlastnosti přímo u vybraného elementu. Soubor s definovanými styly s názvem „format.css“ je obsažen na přiloženém cd.

4.2 PHP a spol

S pomocí značkovacího jazyka HTML, resp. XHTML můžeme sice vytvořit pěkné internetové stránky poskytující množství různých informací, ale pokud chceme aby stránky byly dynamické, např. byly schopny provést nějaké výpočty, spolupracovat s databází nebo mít zabezpečený přístup pomocí hesla, musíme použít další prostředky.

4.2.1 CGI

Jednou z možností jak zavést dynamiku do WWW stránek je CGI – Common Gateway Interface). Program na serveru zpracuje CGI skript a webovému serveru navrátí statickou WWW stránku, která je následně vrácena klientovi jako výstup jeho požadavku. Pro CGI skripty není předepsán konkrétní programovací jazyk. Skripty lze psát například v PERLu, C nebo C++.

4.2.2 ASP

Další alternativou je vytváření dynamických stránek pomocí skriptovacího jazyka ASP (Active Server Pages). Tento jazyk byl vyvinut firmou Microsoft pro operační systémy Windows. Toto prostředí umožňuje vkládat do běžných HTML stránek speciální kódy umožňující funkce, které pomocí HTML nevytvoříme. Pro možnost použití ASP skriptů musí být na serveru nainstalovaná programová podpora ASP. Oproti níže uvedenému PHP je ASP mnohem náročnější na výkon serveru na kterém běží.

Také lze použít ASP.NET jako součást vývojové platformy .NET. Oproti klasickému ASP je ASP.NET kompilované. Při prvním spuštění se aplikace zkompiluje a poté běží mnohem rychleji než klasické interpretované ASP.

4.2.3 PHP

Mezi skriptovací jazyky se řadí také PHP (PHP:Hypertext Preprocessor). Stejně jako ASP patří mezi tzv. server-side skript languages, tzn. jazyky pro vytváření skriptů zpracovávaných na straně

serveru. Komunikace mezi klientem a serverem vypadá takto: Klient (uživatel u počítače) se připojí na server obsahující požadovanou stránku. Stránka se klientovi zobrazí čistě jako HTML, i když obsahuje kód v PHP. Klient může posílat na server různé požadavky které se pomocí programové podpory PHP zpracují na přímo na serveru a ke klientovi se dostane vždy pouze HTML kód. Díky tomu je zpracování PHP nezávislé na internetovém prohlížeči klienta a klient nepotřebuje mít na svém počítači žádnou podporu pro zpracování PHP skriptů.

PHP kód se může psát přímo do souboru s HTML, jeho začátek i konec musí být musí být označen (např. <?php nějaký PHP kód ?>). Soubor obsahující php musí mít příponu php.

Výhody PHP:

- Je multiplatformní – lze jej provozovat na většině dnes existujících operačních systémů
- Je rychlé
- Snadno komunikuje s databázemi jako je MySQL, PostgreSQL MSSQL a řadou dalších
- Patří mezi open-source projekty, je volně šiřitelé
- Neustále se vyvíjí
- Podporuje širokou řadu souvisejících formátů, technologií a standardů, obsahuje řadu vlastních funkcí
- V praxi hojně používané, proto existuje velké množství návodů a manuálů pro jeho popis

4.2.4 JavaScript

V některých souborech jsem také použil JavaScript. Jedná se o tzv. client-side skript language, tzn. že kód napsaný v tomto jazyce je ze serveru poslán společně s HTML ke klientovi a tam se teprve vykonává. Je tedy závislý na internetovém prohlížeči klienta a jeho nastavení. JavaScript se zapisuje přímo mezi HTML kód, používá se např. k vytvoření a ovládání různých interaktivních prvků (tlačítka, textová políčka), ovládání oken, efekty obrázků apod. Javascript je multiplatformní.

4.3 Systémy řízení báze dat

Dříve byla všechna data potřebná k fungování aplikace uložena přímo v programu, později se přesunula do souborů mimo aplikaci. to však mělo mnoho nevýhod, jako opakující se data (redundance), nebezpečí nekonzistence, problémy s neplánovanými dotazy nebo obtížná implementace integritních omezení. Proto se koncem 60. let 20. století data začala ukládat do databází. Databáze komunikují s aplikacemi pomocí systému řízení báze dat (SŘBD). Existují různé modely pro reprezentaci struktury dat v databázi, např. síťový, hierarchický, objektově relační nebo v současné době nejrozšířenější – relační. V relačních databázích jsou data na logické úrovni

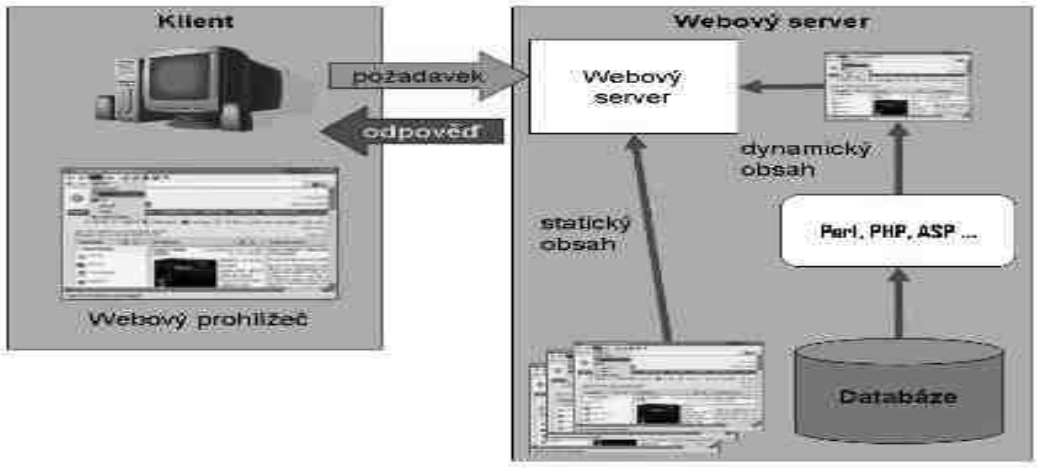
uspořádána do tabulek (relací), nad kterými jsou definovány přípustné operace. Pro ovládání databází používají systémy řízení báze dat obvykle jazyk SQL (Structured Query Language – strukturovaný dotazovací jazyk). Existuje celá řada kvalitních programů pro práci s databází, mezi nejznámější patří MySQL, PostgreSQL, Oracle, MS SQL, Sybase, Informix… Některé z nich (Oracle, MS SQL) jsou vyvíjeny pro komerční využití a jejich cena se pohybuje v desítkách i stovkách tisíc korun, některé jsou dostupné jako freeware (PostgreSQL, pro nekomerční využití MySQL).

Z výše uvedených jsem si vybrat MySQL. Tento databázový systém je velmi oblíbený a hojně využívaný, často se používá v kombinaci s PHP. Je snadno implementovatelný, multiplatformový (pro Windows, Linux, ale i další operační systémy). Mezi další výhody patří jeho rychlost. Existuje k němu velké množství návodů, popisů a manuálů.

Ke správě databáze na svém počítači jsem si nainstaloval program PHPMyAdmin, který umožňuje ovládat MySQL DB server přes webové rozhraní. Pomocí tohoto programu lze jednoduše manipulovat s celými databázemi i jednotlivými tabulkami.

4.4 Webový server Apache

Webový server je program, který pomocí http protokolu odpovídá za vyřizování požadavků od klientů (webovým serverem se také označuje počítač na kterém tento program běží). Klienti posílají své požadavky pomocí svých webových prohlížečů. Vyřízení požadavků znamená v tomto případě zpracování PHP kódu, zpracování příkazů pro komunikaci s databází a odeslání údajů ve formátu webové stránky klientovi. Princip komunikace je znázorněn na následujícím obrázku (obr. 10 – převzato z http://cs.wikipedia.org/wiki/Soubor:Webserver.jpg).



Obr. 10 – Znázornění komunikace klienta s webovým serverem.

Nejrozšířenější web server současnosti je program Apache HTTP Server (http://httpd.apache.org/), který je vyvíjen jako open-source pro systémy Unix a Windows. Pro testování programové podpory diplomové práce na svém počítači jsem si vybral právě tento program.

4.5 Testovací konfigurace

Pro vývoj a testování Programové podpory dokumentace řízení procesů jsem použil tyto verze programů:

- PHP 5.2.0
- PHPMyAdmin 2.6.2
- MySQL 4.1.22
- Apache HTTP Server 2.0.54

Pro tvorbu aplikace jsem také využil HTML, CSS a Javascript. Pro editaci souborů jsem používal program PHPEditorIDE, zejména kvůli zvýraznění syntaxe. Aplikaci jsem testoval na PC pod operačním systémem Windows XP a v internetových prohlížečích Mozilla Firefox a Internet Explorer a na počítači Mac pod operačním systémem Mac OS X Tiger 10.4.09 (http://www.apple.com/macosx/tiger/) s internetovým prohlížečem Safari (http://www.apple.com/macosx/features/safari/). Internet Explorer a Mozilla Firefox jsou v současné době nejpoužívanější internetové prohlížeče pro platformu PC.. Internet Explorer je součástí operačního systému Windows a jeho aktuální verze jsou dostupné na internetu, Mozilla Firefox je také volně dostupný na http://www.mozilla.com.

5 Programová podpora dokumentace procesů

5.1 Specifikace požadavků

V této části jsou upřesněny a rozšířeny a upřesněny údaje ze specifikace požadavků uvedených v kapitole 4 a jsou zde definovány pojmy používané v projektu.

5.1.1 Rozsah systému, vývojové prostředí

Systém bude vytvořen jako aktivní webové stránky bez napojení na jakékoliv okolní systémy. V případě použití v praxi bude umístěn na intranetovém serveru firmy nebo na některém z internetových serverů. K práci se systémem je potřeba internetový prohlížeč Mozilla Firefox nebo Internet Explorer pro PC nebo Safari pro MAC, které musí být povoleny Cookies a JavaScript. Uživatelské prostředí musí být jednoduché na ovládání, s přiměřeně rychlou odezvou a musí umožňovat víceuživatelský přístup. Uživatelské rozhraní by mělo být přehledné a jednoduché na ovládání. Rozhraní pro uživatele bule vytvořeno v českém jazyce. Údaje do systému budou vkládány ručně z klávesnice nebo importovány ze souborů.

Systém bude zabezpečen proti neoprávněnému přístupu pomocí hesla zadávaného při přihlášení.

5.1.2 Definice

5.1.2.1 Uživatel systému (zaměstnanec)

Uživatel je každá osoba pracující se systémem. Může to být zaměstnanec firmy řízené tímto systémem nebo člověk z vývojářské firmy starající se o správu a údržbu systému. Každý z uživatelů je jednoznačně identifikován přihlašovacím jménem – loginem. V databázi jsou o každém uživateli uloženy tyto informace:

- login
- jméno
- příjmení
- heslo

- adresa
- telefon
- email
- role
- informace jestli je blokován

Všechny informace musejí být zadány. Login společně s heslem slouží k přihlášení do systému, společně s rolí určují práva přístupu k jednotlivým částem systému. Uživatel může být v roli:

- správce systému (má přístup do všech částí systému, vidí všechny projekty, procesy, hodnocení i zaměstnance, má právo změnit jakýkoliv údaj). Tento uživatel byl přidán dodatečně, převzal na sebe některé činnosti manažera projektu.
- manažer procesů (spravuje knihovnu procesů, vytváří nové procesy, edituje stávající a sleduje jejich hodnocení)
- manažer projektu (vytváří a spravuje projekty, sleduje průběh procesů v projektu, může zadávat jejich hodnocení)
- přidělený pracovník (pracuje podle jednotlivých procesů, vidí projekty a procesy ve kterých je angažován, může zadávat hodnocení procesů podle kterých pracuje). Při použití systému v praxi by mohla být role přiděleného pracovníka více upřesněna a rozdělena např. na testera, implementátora, grafika apod.

Pokud bude v následujícím textu uvedeno správce systému, manažer procesů, manažer projektu nebo přidělený pracovník, je tím myšleno uživatel systému v této roli.

Každý z uživatelů má v databázi uloženu informaci jestli je blokován. Blokovaný uživatel nemá přístup do systému. Blokování uživatele může být použito např. při odchodu pracovníka z firmy. Zaměstnanci mohou měnit své osobní údaje kromě loginu, role a informace o blokování, zaměstnanec v roli správce systému může měnit všechny údaje kromě loginu sobě i všem ostatním uživatelům. Zaměstnance nelze odstranit z databáze.

5.1.2.2 Projekt

Součástí systému bude databáze všech projektů firmy, obsahující informace o uskutečněných, rozpracovaných i naplánovaných projektech. O každém projektu musejí být uloženy tyto informace:

- název
- login pracovníka který ho vytvořil
- termín zahájení prací
- termín ukončení prací
- stav projektu
- seznam procesů podle kterých je projekt nebo jeho produkt vytvářen

Všechny informace musejí být zadány, přičemž datum zahájení a zejména ukončení projektu lze snadno změnit v závislosti na vývoji projektu. V projektu půjde také mazat a přidávat jednotlivé procesy. Projekt může založit zaměstnanec v roli manažera projektu nebo správce systému. Uživatelé v těchto rolích i manažer projektu vidí všechny procesy a informace o nich, přidělený pracovník vidí pouze procesy ve kterých je angažován. Stav projektu je uzavřen/neuzavřen.

5.1.2.3 Proces

Procesy znázorňují posloupnost činností které je potřeba provést abychom získali nějaký výstup ze zadaných vstupů. V systému bude obsažena knihovna procesů obsahující různé procesy. O každém procesu budou k dispozici tyto údaje:

- název
- login pracovníka který ho vytvořil
- obrázek (diagram) znázorňující tento proces
- textový popis vysvětlující a upřesňující cíl procesu a práce na něm. V popisu budou také uvedeny vstupy a výstupy procesu
- typ procesu, tzn. v které části vývoje projektu ho lze použít (např. analýza, implementace, řízení nákladů…)

V případě že je proces použit jako součást nějakého projektu, budou k dispozici ještě další údaje, vázané ke konkrétnímu projektu:

- termín zahájení činností
- termín ukončení činností
- seznam zaměstnanců pracujících na procesu
- stav procesu
- dokumenty a data sloužící jako vstupy a výstupy procesu

Proces může vytvořit manažer projektu nebo správce systému. Manažer procesů nemá k projektům přístup, přidělený pracovník vidí pouze projekty ve kterých je angažován. Diagramy k procesům budou uloženy na disku ve formě obrázků typu bmp, gif, jpeg nebo png, uživatelé budou mít možnost nahrát na server nový soubor nebo smazat stávající. Všechny údaje o procesy musí být zadány. Kromě názvu a loginu autora bude možnost údaje editovat. Stav procesu může být: nezahájen, rozpracován, ukončen. Stav procesu mění manažer projektu nebo správce systému. Proces lze smazat jak z projektu tak databáze, proces nemůže být smazán z databáze pokud je součástí nějakého projektu. Dokumenty budou nahrávány na server může s nimi manipulovat i přidělený pracovník (pouze v procesech ve kterých je angažován).

5.1.2.4 Hodnocení

Ke zdokonalení procesu a jeho efektivnější použití v dalších projektech musí být procesy nějakým způsobem měřeny, vyhodnocovány a podle toho následně zlepšovány. Podle naměřených údajů lze také zjistit jestli je proces stabilní (viz kapitola 2.3.4.1).

U každého procesu v konkrétním projektu bude moci přidělený pracovník, vedoucí projektu nebo správce systému vkládat své připomínky a nápady na zlepšení ve formě textového popisu zadávaného z klávesnice.

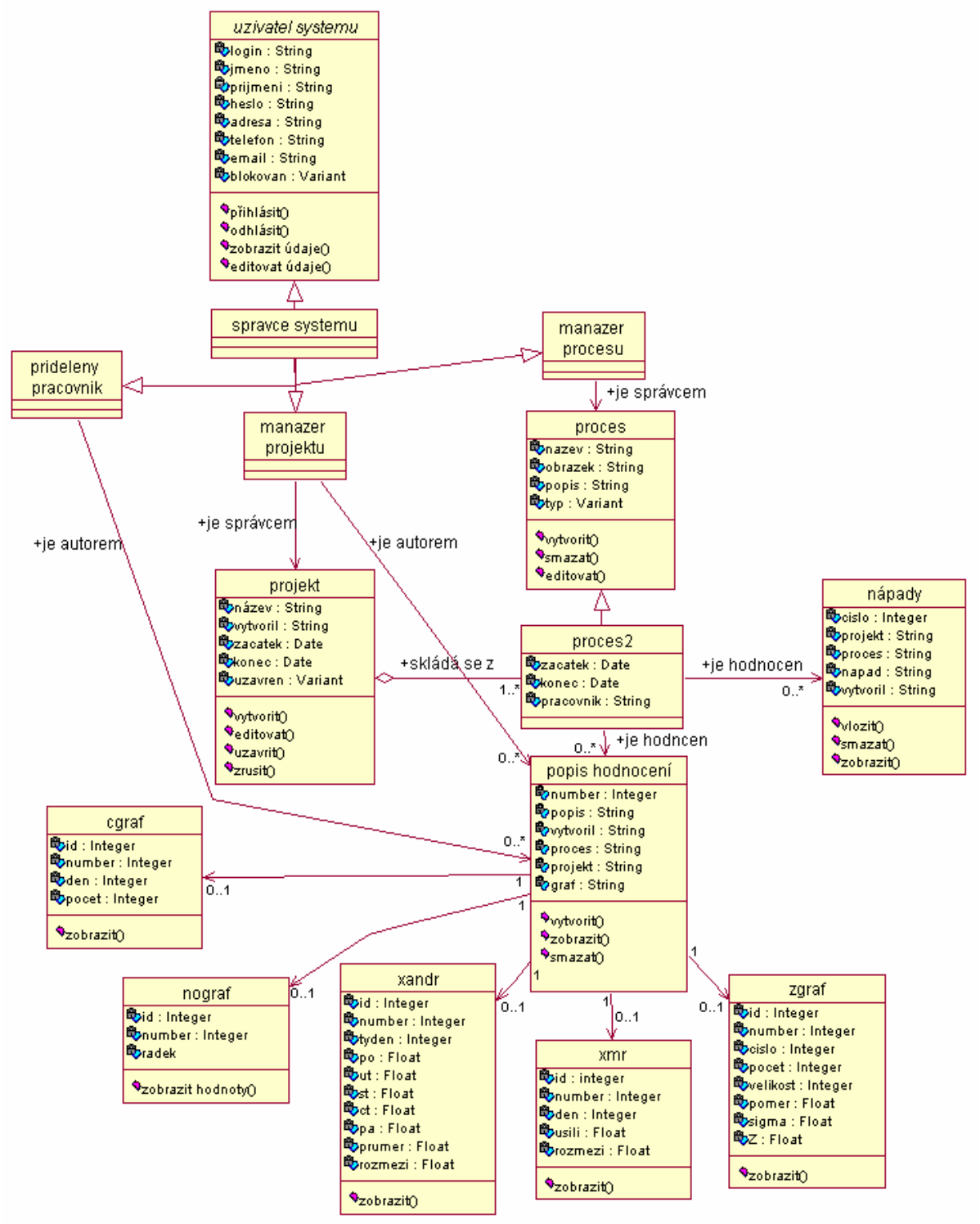
Další hodnocení nebo měření se bude vkládat ze souboru ve formátu XML v předem definovaném formátu. Data budou zpracována a zobrazena pomocí grafů uvedených v kapitole 2.3.4.2. Při každém vložení dat ze souboru musí pracovník připojit popis, ve kterém specifikuje jaká data byla vložena a co bylo měřeno. Pracovník může také z XML souboru vkládat data, která nebudou zobrazována žádnými grafy. Kromě zobrazení grafů si uživatel bude moci prohlédnout i hodnoty ze kterých byl graf vytvořen.

Zobrazení hodnot a grafů bude možno přes projekt v němž jsou jednotlivé procesy a zde bude možnost také zadávat naměřená data. Další možností bude zobrazení všech procesů v databázi i s hodnoceními, zde nebude možnost vkládat žádné údaje. Uživatel v roli přiděleného pracovníka uvidí pouze procesy ve kterých je angažován, ostatní uživatelé uvidí všechny procesy.

Ke každému procesu v určitém projektu bude také možnost vložit jakýkoliv soubor s daty sloužící jako vstup/výstup procesu.

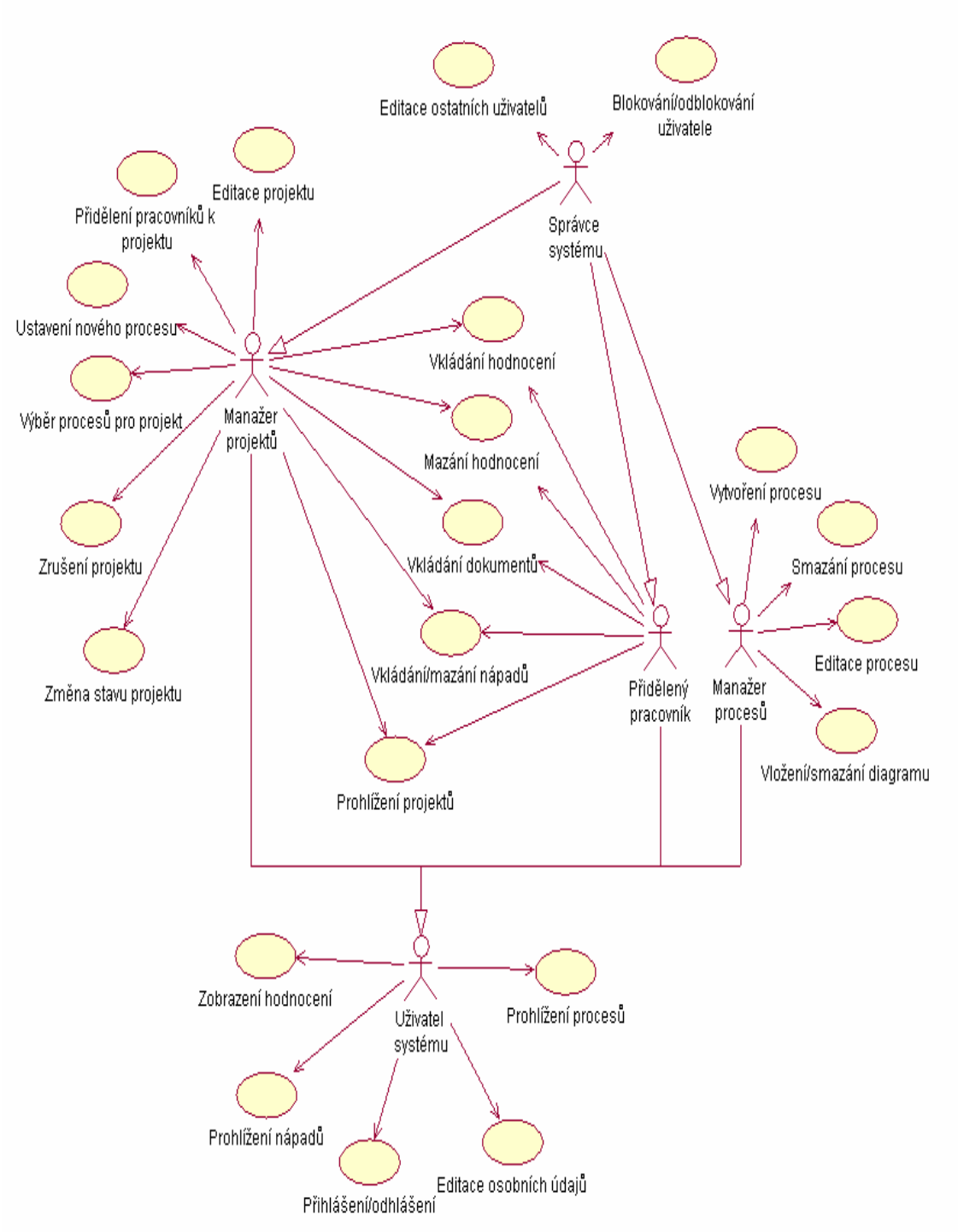
5.2 Návrh systému

Při návrhu systému jsem vycházel ze specifikace požadavků (kapitola 5.1). Využitím jazyka UML(Unified Modeling Language) a programu Rational Rose firmy Rafional Software (nyní pod IBM - http://www-306.ibm.com/software/rational/) jsem vytvořil Class diagram ukazující celkový konceptuální pohled na systém (obr. 11).



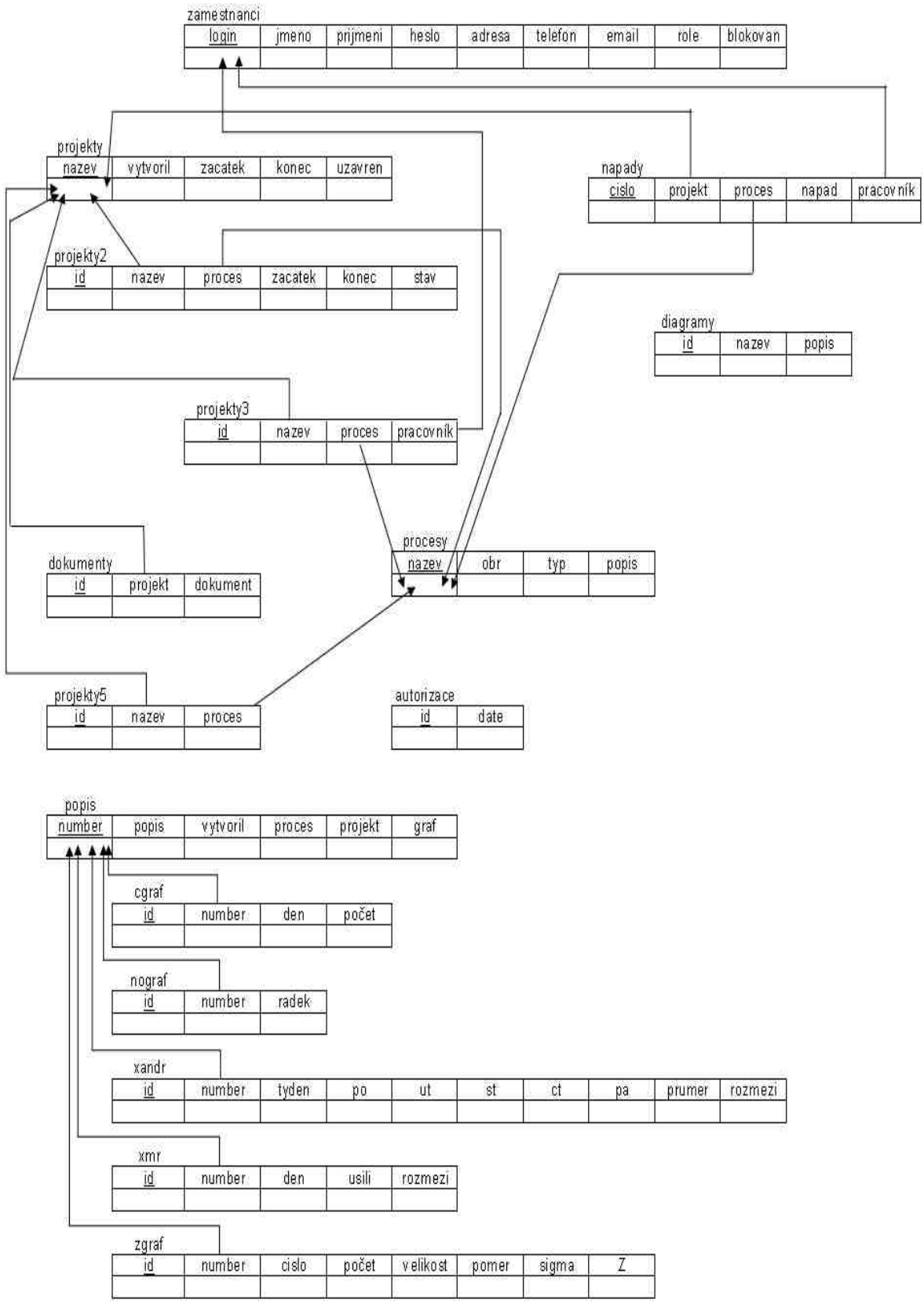
Obr 11. Class Diagram

Dále jsem vytvořil diagram použití – UseCase diagram (obr. 12), znázorňující jednotlivé uživatele systému v jejich rolích a případy použití, to znamená činnosti které může uživatel vykonávat. Uživatel v roli správce systému může vykonávat všechny činnosti, uživatelé v ostatních rolích mohou vykonávat pouze činnosti které jsou uvedeny u jeho role. Aktér v roli Uživatel systému ve skutečnosti neexistuje, pouze znázorňuje případy použití které mohou využívat uživatelé ve všech rolích (kvůli přehlednosti diagramu).



Obr. 12 UseCase Diagram

Nakonec jsem vytvořil datový model aplikace, znázorňující tabulky pro relační databázi MySQL a jejich propojení (obr. 13).



Obr. 13 Datový model aplikace

Tabulka „zamestnanci“ obsahuje údaje o uživatelích systému podle specifikace systému, stejně tak v tabulkách „projekty“ a „procesy“ jsou uloženy základní hodnoty o projektech, resp. procesech v databázi.

V tabulkách „projekty2“, projekty3“ a „dokumenty“ jsou uvedeny informace ke konkrétním projektům. V „projekty2“ jsou procesy podle kterých je projekt vytvářen, jejich data zahájení a ukončení a jejich stav, v „projekty3“ jsou uživatelé v roli přidělených pracovníků pracující na jednotlivých procesech v projektu a v „dokumenty“ jsou informace o dokumentech(vstupech a výstupech) procesů. Tabulka „projekty5“ se používá jako pomocná při vytváření projektu a neobsahuje žádné perzistentní informace.

Do tabulky „autorizace“ se ukládá hodnota session získaná při přihlášení a datum posledního přístupu k systému.

„Diagramy“ obsahují názvy obrázků, dostupných ke znázornění procesů uložených ve složce procesy aktuálního adresáře aplikace, a jejich popis.

Do ostatních tabulek se ukládají údaje o hodnocení procesů.

Systém bude tvořen ze čtyř modulů:

- Projekty – v tomto modulu bude možnost vytvořit nový projekt, smazat jej, editovat, uzavřít a prohlížet, dále přidat/smazat/editovat proces patřící do projektu a údaje s tím související (začátek a konec procesu, pracovníky přidělené k procesu). U každého projektu bude automaticky uložen jeho autor (přihlášený uživatel).
- Knihovna procesů – zde bude možnost zobrazit všechny stávající procesy, dále vytvářet nové, editovat je nebo mazat. Také je v tomto modulu možnost nahrát na server diagram patřící k procesu nebo smazat diagram z databáze. U každého projektu bude malý náhled diagramu, po kliknutí na tento náhled se v novém okně zobrazí diagram v původní velikosti. U každého procesu bude automaticky uložen jeho autor (přihlášený uživatel).
- Hodnocení procesů – v této části se budou hodnotit jednotlivé procesy. Nejdříve se zobrazí projekty v databázi, po kliknutí na název projektu se zobrazí jeho procesy. Uživatel bude moci měnit stav projektu, nahrávat na server/stahovat do počítače dokumenty sloužící jako vstupy nebo výstupy procesu, vkládat/mazat své připomínky a nápady a také vkládat ze souboru údaje pro hodnocení procesů. Výsledek hodnocení bude v některých případech možno zobrazit pomocí některého z grafů: X-bar a graf rozmezí, XMR graf a graf rozmezí, c graf nebo z graf. dále bude v tomto modulu zobrazit všechna hodnocení jednotlivých procesů ze všech projektů, přičemž nebude možnost vkládat nové hodnocení (to je vždy vázáno k nějakému projektu). U každého hodnocení i vloženého nápadu bude automaticky uložen jeho autor (přihlášený uživatel).
- Editace uživatele – v tomto modulu bude moci každý uživatel měnit své osobní údaje kromě loginu, role a informace o tom jestli je blokován, uživatel v roli správce systému může

editovat všechny údaje u všech uživatelů. Správce může uživatele jehož údaje chce editovat vybrat ze seznamu nebo vyhledat pomocí formuláře. Při editaci údajů není potřeba vkládat heslo, při změně hesla však musí být zadáno dvakrát kvůli ověření. Správce systému také může vytvořit nového uživatele. Uživatele nelze smazat, informace o něm musí zůstat např. kvůli hodnocení které vytvořil.

5.3 Implementace a návod pro uživatele

V této části se zaměřím na systém z pohledu uživatele s popisem důležitých částí z pohledu implementace.

5.3.1 Uživatelské rozhraní

Vzhled uživatelského rozhraní jsem vytvořil pomocí grafického editoru Photoshop firmy Adobe (vytvoření a úprava jednotlivých obrázků) a HTML a CSS (zobrazení a umístění na stránce). Přihlašovací stránka je na obrázku obr. 14, příklad hlavní stránky systému na obrázku obr. 15. Přihlašovací stránka je uložena v souboru index.php (jednoduchý popis všech souborů se skripty je uložen v souboru popis.txt na přiloženém cd). Na přihlašovací stránce je zobrazeno logo a formulář sloužící k přihlášení.

Po úspěšném přihlášení se objeví úvodní stránka systému, umožňující přístup k jednotlivým modulům systému. V hlavičce stránky je logo aplikace, pod ním tlačítka fungující jako odkazy k jednotlivým modulům, k přechodu na úvodní stránku nebo umožňující odhlášení uživatele. V levém sloupci jsou během celého přihlášení uvedeny údaje o uživateli – jeho login a role v systému a také aktuální datum. V pravém sloupci se zobrazují informace jednotlivých modulů.

Základ hlavní stránky je v souboru design.php, do které se závislosti na pohybu uživatele v systému vkládají pomocí php příkazu INCLUDE jednotlivé stránky (např. INCLUDE “info.php“). Pro jednotlivé elementy stránek (formuláře, tabulky, jednotlivé druhy textu..) jsem vytvořil soubor se styly format.css. Např. styl pro tlačítko formuláře vypadá takto:

```
.logbut {
background-color: #B13028;
text-align: center;
color: white;
font-weight: bold;
font-family: Verdana, Arial,  Helvetica, sans-serif;
font-size: 8pt;
```

cursor: pointer;

}

Některé prvky jsou ještě dále formátovány v HTML pomocí atributů jednotlivých prvků.

Obr. 14 – Přihlášení do systému

Obr. 15 Grafické rozhraní systému

K úpravě designu stránek používám i JavaScript, např. ke změně barvy tlačítek na liště po přejetí kurzorem myši (v tomto případě znamená změna barvy změnu obrázku):

```
<a href="design.php?str=uvod"><img src="obrazy/04b_02.gif" onmouseover="this.src
   ='obrazy/04b_02over.gif';" onmouseout="this.src='obrazy/04b_02.gif';"></a>
```

Pomocí JavaScriptu také zobrazuji tlačítka která používám jako odkazy na stránkách, v příkladu je uvedeno tlačítko Zpět vedoucí :na stránku 'design.php?str=proj'.

```
<input type="button" value="Zpět" class="logbut" onClick = "window.location =
 'design.php?str=proj'">
```

Veškerá tlačítka Zpět jsou zpracována jako odkazy na určitou stránku a přesunou uživatele systému o úroveň výš/zpět. Z toho důvodu mají tlačítka Zpět všechny stránky kromě úvodních stránek jednotlivých modulů.

Uživatelské rozhraní systému je kompletně v češtině, všechny stránky mají nastavenou znakovou sadu iso-8859-2.

5.3.2 Přihlášení uživatele

Přístup k systému je chráněn uživatelským jménem (loginem) a heslem. Každý uživatel má přiděleny tyto údaje, kterými se musí prokázat při přihlášení, jinak je mu přístup odepřen. Systém také nepovolí přístup blokovanému uživateli. Při přihlašování jsou údaje porovnávány s údaji v databázi, pokud si odpovídají, je uživatel přesměrován na úvodní stránku systému, jinak se mu zobrazí hlášení o neoprávněném přístupu. Při úspěšném přihlášení do systému je vytvořena session a její číslo je uloženo do databáze společně s datem (resp. časem) jejího vytvoření. Tato session provází uživatele celou dobu práce se systémem a je zrušena pouze při odhlášení nebo zavření okna prohlížeče. Její číslo je kontrolováno a porovnáváno z databází při přístupu na každou stránku, takže nikdo kdo nesplnil podmínky přihlášení nemá šanci dostat se do systému. Při přístupu na stránky je také testována doba posledního přístupu, pokud je někdo neaktivní více jak 1 hodinu (3600 sekund), systém ho automaticky odhlásí (zablokuje přístup).

Pro zvýšení bezpečnosti jsem při ukládání hesla použil hashovací funkci MD5, jež je součástí jazyka php. Při každém uložení hesla je zadaný řetězec převeden na hash o délce 16 bytů. To se děje i při přihlašování před porovnáním s databází. Hashovací algoritmus MD5 však není odolný proti kolizím. Kdyby útočník získal zašifrované heslo z databáze a našel k němu kolizi, mohl by kolizní

řetězec použít a dostat se do systému. Proto k heslu před zašifrováním přidám další řetězec, který útočník nezjistí a kolize mu tak nepomůže.

```
$heslo2=$heslo."5q4xw6";
$password=MD5($heslo2);
```

Existuje ještě možnost odposlechu hesla při předávání ve formuláři. Tomu by se dalo zabránit použitím šifrovaného spojení, např. SSL.

5.3.4 Modul projekty

K této části mají přístup uživatelé v roli správce systému, přiděleného pracovníka a manažera projektu. Manažer procesů nemá k modulu přístup. Práva přístupu jsou řešena voláním php skriptu při vstupu do modulu, kde se zjistí role uživatele a ten je potom přesměrován na stránky zobrazující informace příslušející jeho roli. Konkrétně při vstupu do modulu projekty je volán skript str_proj.php. Ten manažera procesů odkáže na str_proj0.php (zobrazí hlášení o odmítnutí přístupu), přiděleného pracovníka odkáže na str_proj1.php a uživatele v rolích správce systému a manažera projektů odkáže na str_proj2.php.

Manažer projektu nebo správce systému může kliknutím na tlačítko „Nový projekt“ vytvořit nový projekt. Po kliknutí na tlačítko se objeví formulář pro zadání údajů. Je nutné zadat název projektu, datum začátku a datum konce projektu ve formátu rrrr-mm-dd a také je možné označit procesy které chceme zahrnout do projektu. Název projektu nesmí být stejný s jiným projektem v databázi, datum musí být v zadaném formátu, musí existovat a počáteční datum nesmí být později než datum ukončení. V opačném případě dojde k výpisu příslušného hlášení. Při práci na projektu může dojít k posouvání termínů, proto mohou být data zadána pouze orientačně a je možnost je kdykoliv změnit. Při vložení korektních údajů a stisknutí tlačítka uložit se nabídne možnost pokračovat a zadat tak informace ke zvoleným procesům, kliknutím na tlačítko zpět se uživatel dostane na výpis procesů. Názvy procesů zvolených uživatelem se uloží do tabulky „projekty5“, odkud se čtou a po korektním zadání údajů mažou. V případě že uživatel zvolí pokračovat, může ke každému procesu zadat datum začátku a konce, přičemž platí stejná kritéria jako u projektu a navíc data procesu musí být v rozmezí dat projektu. Také je zde možnost zadat až 10 přidělených pracovníků. Všechny hodnoty lze později změnit.

Na úvodní straně modulu je tabulka s jednotlivými projekty, seřazenými podle abecedy, přičemž se nejdříve vypíší neuzavřené projekty. Správce systému vidí všechny projekty v databázi, manažer projektů jen ty které vytvořil a přidělený pracovník pouze ty ve kterých je angažován. U procesu je uvedeno jméno autora, po kliknutí na něj vyskočí emailové okno s emailovou adresou tohoto

pracovníka. Tímto způsobem je možné komunikovat v celém systému (vždy když je někde uvedeno jméno pracovníka, funguje jako odkaz na napsání emailu). Tyto odkazy jsem vytvořil pomocí JavaScriptu:

```
<a href="mailto:'.$radek[0].'" class="logtext" style="text-decoration:none;
 color: #181318;">'.$radek2[0].'</a>
```

kde v proměnné $radek[0] je uložen email a v proměnné $radek2[0] login uživatele.

Správce systému a manažer projektu navíc mohou uzavřít/obnovit nebo smazat projekt. Před smazáním projektu se systém zeptá uživatele jestli chce projekt opravdu smazat.

Po kliknutí na jméno projektu se přidělenému pracovníkovi zobrazí informace o procesech v projektu ke kterým byl přidělen. Ostatní vidí informace o všech procesech v projektu. Také mohou editovat termíny projektu, smazat proces nebo přidat nový. Po kliknutí na jméno procesu se zobrazí údaje o procesu vzhledem k projektu, je možné je měnit.

5.3.4 Modul knihovna procesů

K tomuto modulu mají přístup všichni uživatelé. Uživatel v roli přiděleného pracovníka vidí procesy ve kterých je zapojen, manažer projektů všechny procesy. Po kliknutí na malý náhled diagramu v tabulce se diagram zobrazí v novém okně v původní velikosti. Zobrazení obrázku je realizováno voláním skriptu diagram.php do nového okna:

```
echo '<a href="diagram.php?obr='.$radek[1].'" target=_blank>'."
<img src=\"".$radek[1]."\"style=\"width:50px; height:50px; border:0px\">".'</a>';
```

zobrazí náhled obrázku jako odkaz a zajistí jeho otevření v novém okně,

```
echo  "<img src=\"".$adresa."\"style=\"padding:0px; margin:0px; border:0px;\">";
```

zobrazí obrázek v původní velikosti. V proměnných jsou názvy souboru s diagramy.

Ostatní uživatelé mají možnost proces smazat nebo vytvořit nový. Při vytváření nového procesu je nutné zadat název (kontroluje se jestli už neexistuje proces se stejným jménem), jméno diagramu s grafickým znázorněním procesu, slovní popis procesu obsahující vstupy a výstupy procesu. Také se z nabídky vybírá typ procesu, ukazující ve které fázi projektu lze proces použít. Kliknutím na název procesu ve výpisu systém přesměruje uživatele na stránku umožňující jeho editaci . Ve formuláři jsou předvyplněny aktuální údaje o procesu.

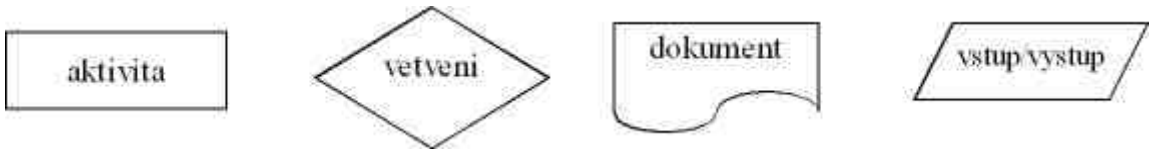
Manažer procesů a správce systému také nahrát na server nový diagram nebo smazat stávající. Soubor s obrázkem se zadává prostřednictvím formuláře a je předáván metodou POST, kódování je nastaveno na enctype="multipart/form-data". Soubor se po odeslání formuláře automaticky zkopíruje do složky Temp ve Windows. Nejdříve zjistíme jméno souboru, celou cestu k němu, velikost a typ:

```
$fileName = $_FILES['userfile']['name'];
$tmpName  = $_FILES['userfile']['tmp_name'];
$fileSize = $_FILES['userfile']['size'];
$fileType = $_FILES['userfile']['type'];
```

„userfile“ je jméno označující soubor poslaný formulářem. Velikost souboru nesmí přesáhnout 2MB, soubor musí být typu image/gif, image/bmp, image/png nebo image/jpeg. Soubor se uloží do složky procesy v aktuálním adresáři (ten ve kterém jsou skripty systému). Soubor se ukládá pod svým jménem nebo je možno zadat jiný název. Pokud ve složce již existuje soubor se stejným názvem, objeví se upozornění a uživatel musí zadat jiný název. Soubor se ukládá pomocí funkce move_uploaded_file:

```
move_uploaded_file($_FILES['userfile']['tmp_name'], './procesy/'.$fileName)
```

Tato funkce zkopíruje uploadovaný soubor do složky procesy. Když dojde při přesouvání souboru k chybě, funkce vrátí hodnotu false a vypíše se příslušné hlášení. Po nahrání souboru na server se ještě pomocí funkce chmod nastaví přístupová práva k souboru. Pokud vše proběhne správně, vloží se do databáze název souboru společně se zadaným popisem. V diagramech jsou jednotlivé prvky znázorněny značkami uvedenými na obrázku obr.16. Diagramy jsou vytvářeny pomocí programu SmartDraw, domovská stránka viz. [7].



Obr. 16

5.3.5 Modul hodnocení procesů

V této části je možno hodnotit jednotlivé procesy, sledovat hodnocení, zobrazit některá měření pomocí grafů, dále vkládat a stahovat dokumenty vytvořené při práci na procesu (vstupy a výstupy) a také měnit stav procesu. K modulu mají přístup uživatelé ve všech rolích.

Na úvodní stránce jsou vypsány projekty v databázi, kliknutím na název se zobrazí procesy obsažené v projektu. Přidělený pracovník vidí jen projekty a procesy ke kterým byl přidělen, manažer projektů vidí projekty jež vytvořil, manažer procesů rozdělení podle projektů nevidí, správce systému vidí vše.

Manažer projektů a správce systému má možnost měnit stav procesu v projektu. Stav procesu může být „nezahájen“, „rozpracován“ nebo „ukončen“. Ukončenému procesu lze nastavit zpět stav rozpracován.

Všichni uživatelé kromě manažera procesů mohou do systému nahrát dokumenty týkající se procesu. Tyto soubory jsou uloženy do složky „dokumenty“ v aktuálním adresáři, ukládání probíhá obdobným způsobem jako při ukládání obrázků s diagramem procesu. K uložení souboru se uživatel dostane kliknutím na odkaz „stáhnout/nahrát“ v tabulce s procesy ve sloupci „vstupy/výstupy“. Po kliknutí se také vypíší všechny uložené dokumenty jako odkazy, kliknutím na ně je lze prohlížet nebo stáhnout do svého počítače. Správce systému a manažer projektu může dokumenty mazat.

Ve sloupci „nápady“ lze kliknutím na nápis „přidat“ vložit z klávesnice nějakou připomínku nebo nápad týkající se procesu. Pokud už byl v procesu v konkrétním projektu nějaký nápad přidán, zobrazí se v tabulce v příslušné buňce nápis ukázat. Kliknutím na něj se zobrazí všechny nápady patřící k tomuto procesu v tomto projektu. Je zde také možnost nápad smazat.

Hlavním smyslem tohoto modulu je hodnotit proces pomocí naměřených dat. Pro zadávání jednotlivých možností jsem připravil pět různých možností. Z důvodu předpokládaného velkého množství údajů jsou data zadávána ze souborů. Soubory musí být ve formátu xml a údaje v nich musí splňovat předepsanou syntaxi. Příklady jednotlivých xml souborů jsou na přiloženém cd.

Jednou možností hodnocení je zadat libovolná data. Tato data jsou načtena ze souboru a poté uložena do databáze. U každého zadaného měření musí být připojen popis vysvětlující co jednotlivé hodnoty znamenají, případně další údaje o měření (např. podmínky za kterých byla data měřena). Zadaná data nejsou prezentována žádným grafem, uživatel může pouze zobrazit hodnoty a jejich popis.

Pro čtení dat ze souboru jsem vytvořil jednoduchý xml parser v pěti verzích pro jednotlivé typy měření, který zjistí jestli jsou data v souboru ve správném formátu a pokud ano, uloží je do databáze. Funkce „tag1“ kontroluje počáteční tag, zjišťuje zda je správně zapsán a odpovídá tagům které jsou určeny pro tento typ měření a také jestli je správně pořadí jednotlivých tagů. Funkce „tag2“ kontroluje uzavírající tagy. Funkce „uloz“ zpracovává údaje a ukládá je do pole. Nejdříve jsou do pole uloženy všechny údaje ze souboru a pokud při čtení nedošlo k nějaké chybě a data v souboru jsou syntakticky správně, jsou dále zpracovávána (kontroluje se např. jestli v místech kde mají být čísla jsou opravdu čísla a žádné jiné znaky) a ukládána do databáze.

5.3.5.1 Měření reprezentovaná X-bar a R grafem

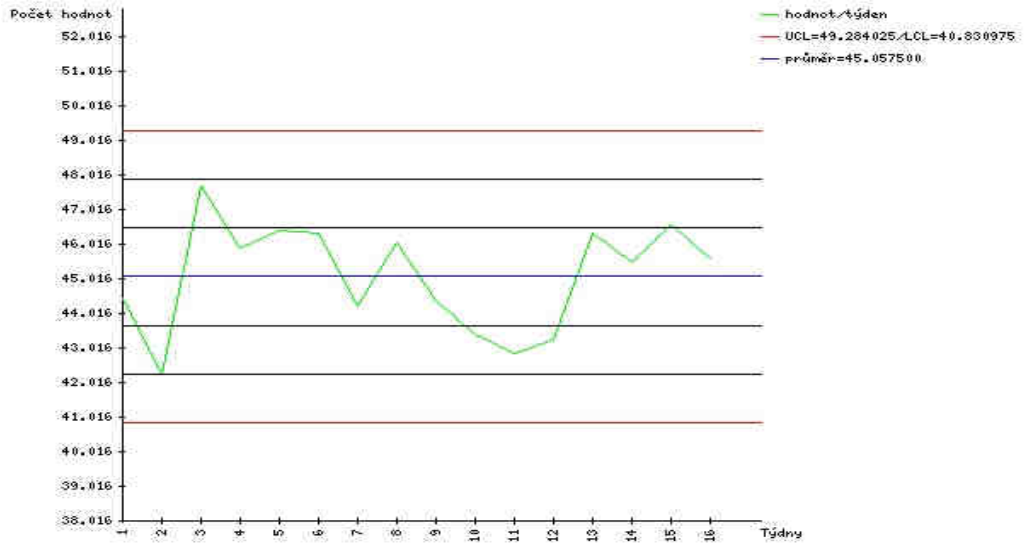
Prvním z měření reprezentovaných grafem je denní měření hodnot, údaje jsou sloučeny do skupin po pěti (po týdnech). Tím se skryjí jednodenní náhodné výkyvy v úsilí a zobrazí se dlouhodobější tendence. Tato měření jsou reprezentována X-bar grafem a grafem rozmezí. X-bar zobrazuje hlavní tendence procesu a změny v průběhu času, graf rozmezí (R graf) ukazuje změny ve skupinách. Toto hodnocení, stejně jako ostatní, slouží jako šablona pro libovolná měření, kdy sledujeme hodnoty po dnech. Můžeme např. měřit počet spotřebovaných člověkohodin/den, počet napsaných řádků kódu, počet otestovaných souborů nebo spotřebované zdroje. K objasnění toho co bylo měřeno slouží popis ke grafu. V X-bar grafu jsou zobrazeny kontrolní meze (horní a dolní mez), linie průměrné hodnoty a také úsečky po 1-sigma intervalech sloužící k určení stability procesu. V grafu rozmezí je zobrazena horní mez a průměrná hodnota. Rovnice pro výpočty jednotlivých hodnot lze najít v souborech grafxbar.php a grafrangexar.php na přiloženém cd nebo v [4].

Příklad (viz [4]): Rozvrh vývoje nového produktu ve firmě je založen na předpokladu, že podpora zabere asi 40 člověkohodin za den, dokud nebude nový produkt vydán. Každodenní úsilí musí zůstat v předpokládaných mezích, jinak bude narušen rozvrh vývoje a bude nutno vytvořit alternativní plány. Hodnoty měření za posledních 16 týdnů jsou v tabulce tab. 1, výsledný X-bar graf na obrázku obr 17 a graf rozmezí na obrázku obr. 18.

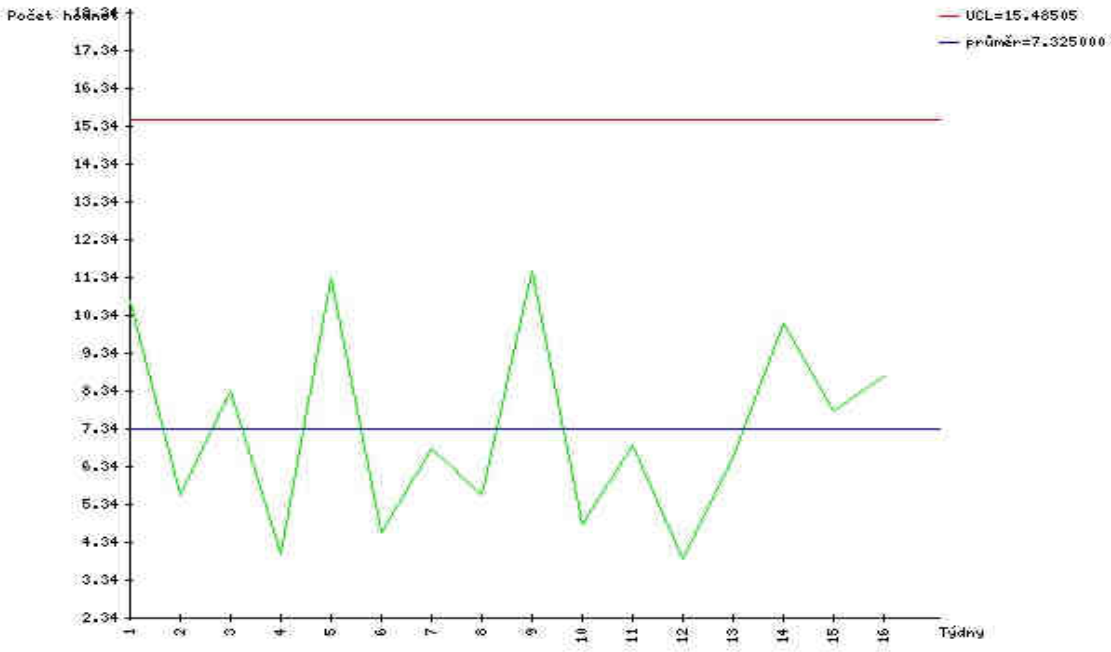
Následující obrázky a tabulky (obr. 17-22, tab. 1-4) jsou vyjmuty z vytvořeného systému programové podpory pro dokumentaci procesů.

Týden	Po	Út	St	Čt	Pá	Průměr	Rozmezí
1	50.50	43.50	45.50	39.80	42.90	44.44	10.70
2	44.30	44.90	42.90	39.80	39.30	42.24	5.60
3	48.80	51.00	44.30	43.00	51.30	47.68	8.30
4	46.30	45.20	48.10	45.70	44.10	45.88	4.00
5	40.60	45.70	51.90	47.30	46.40	46.38	11.30
6	44.40	49.00	47.90	45.50	44.80	46.32	4.60
7	46.00	41.10	44.10	41.80	47.90	44.18	6.80
8	44.90	43.40	49.00	45.50	47.40	46.04	5.60
9	50.00	49.00	42.60	41.70	38.50	44.36	11.50
10	44.50	46.50	41.70	42.60	41.70	43.40	4.80
11	43.80	41.80	45.50	44.50	38.60	42.84	6.90
12	43.20	43.80	44.80	43.50	40.90	43.24	3.90
13	50.00	43.40	48.30	46.40	43.40	46.30	6.60
14	52.30	45.20	42.20	44.80	42.80	45.46	10.10
15	50.00	46.20	47.40	42.20	47.00	46.56	7.80
16	47.30	49.70	48.00	42.00	41.00	45.60	8.70
						45.057500	7.325000

Tab. 1 – Hodnoty pro X-bar graf



Obr. 17 – X-bar graf



Obr. 18 – Graf rozmezí

Z uvedených grafů je vidět, že týdenní průměry i rozmezí jsou v kontrolních mezích. Použitím testů z kapitoly 2.3.4.1 zjistíme že proces neobsahuje žádné známky nestability. Průměrná denní hodnota je přibližně 45 člověkohodin, což je 5 hodin nad plánovanou hodnotu. Toto by mělo být vzato v potaz a měl by být upraven proces podpory nebo upravena kvalita produktu.

5.3.5.2 Měření reprezentovaná XMR a R grafem

Tato měření použijeme pokud chceme použít každou hodnotu měření samu o sobě. Velikost podskupiny je v tomto případě jedna. Je připravený xml soubor (šablona) pro zadávání měření

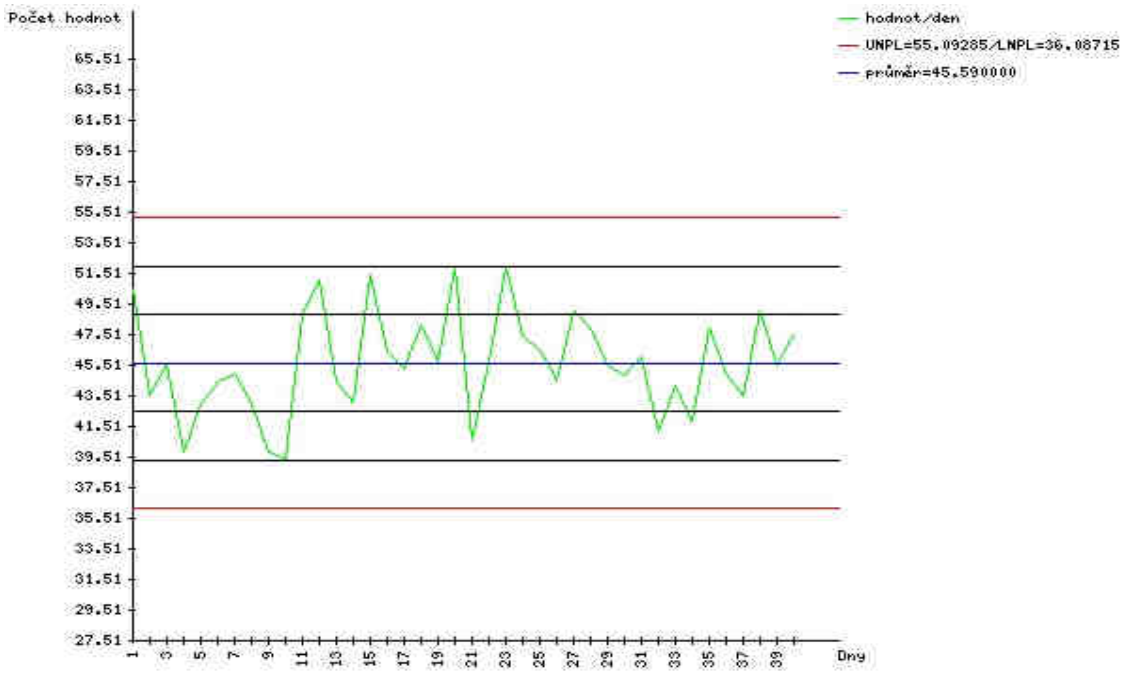
libovolného úsilí z jednotlivých dní. V popisu musí být opět uvedeno co bylo měřeno, eventuelně za jakých okolností. Pokud to uživatel uvede v popisu, nemusí být zadávána měření pouze po dnech, ale např. po hodinách. Rovnice pro výpočty jednotlivých hodnot pro vytvoření grafu lze najít ve [4] nebo v souborem grafxmr.php a grafrangexmr.php.

Příklad (viz [4]): Uživatel má za úkol vyhodnotit dennodenní proměnlivosti úsilí vynaložené na spravování existujícího produktu. Protože jeho hlavním zájmem jsou dennodenní data, použije XMR grafy. Údaje o měřeních v jednotlivých dnech (počet člověkohodin/den) jsou v tabulce tab. 2.

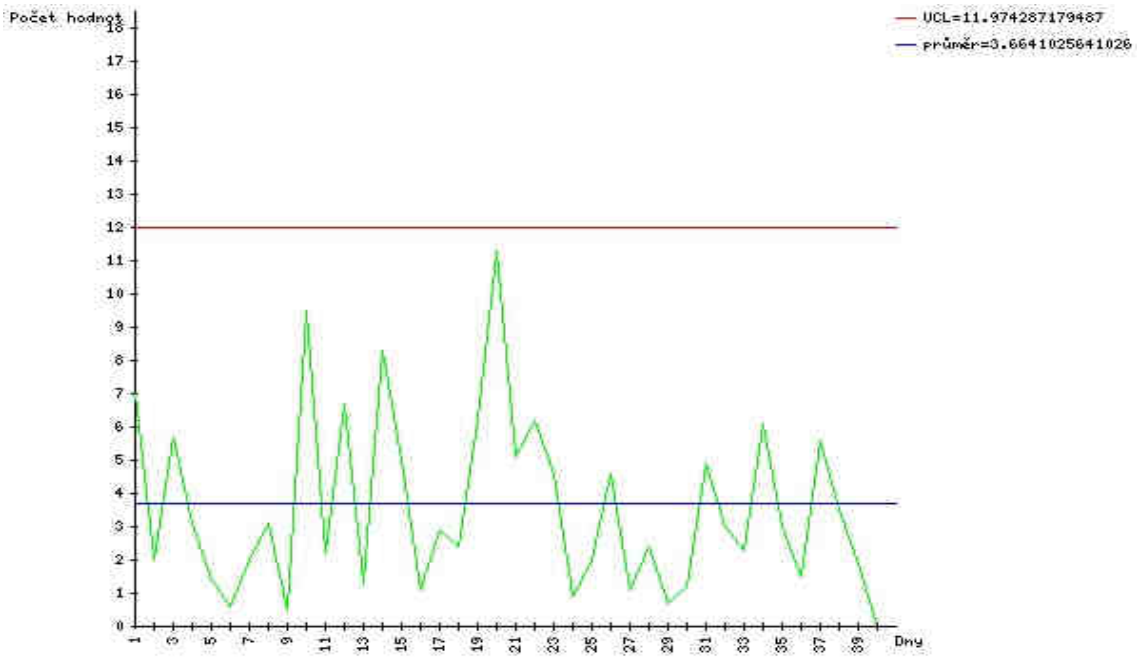
Den	Úsilí	Rozmezí
1	50.50	7.00
2	43.50	2.00
3	45.50	5.70
4	39.80	3.10
5	42.90	1.40
6	44.30	0.60
7	44.90	2.00
8	42.90	3.10
9	39.80	0.50
10	39.30	9.50
11	48.80	2.20
12	51.00	6.70
13	44.30	1.30
14	43.00	8.30
15	51.30	5.00
16	46.30	1.10
17	45.20	2.90
18	48.10	2.40
19	45.70	6.20
20	51.90	11.30
21	40.60	5.10
22	45.70	6.20
23	51.90	4.60
24	47.30	0.90
25	46.40	2.00
26	44.40	4.60
27	49.00	1.10
28	47.90	2.40
29	45.50	0.70
30	44.80	1.20
31	46.00	4.90
32	41.10	3.00
33	44.10	2.30
34	41.80	6.10
35	47.90	3.00
36	44.90	1.50
37	43.40	5.60
38	49.00	3.50
39	45.50	1.90
40	47.40	0.00
	45.590000	3.572500

Tab. 2 – Hodnoty pro XMR graf

Výsledný XMR graf s kontrolními mezemi a úsečkami v intervalech 1-sigma je na obrázku obr. 19, graf rozmezí je na obrázku obr. 20.



Obr. 19 – XMR graf



Obr. 20 – Graf rozmezí

5.3.5.3 Měření reprezentovaná c grafem

Tento druh měření nejlépe využijeme při měření počtu nezávislých událostí, např. při měření počtu chyb objevených během testování nebo kontroly kódu, počtu pádů sytému nebo počtu přístupů k určitému modulu v definovaném časovém rozmezí. V popisu musí uživatel opět upřesnit co přesně

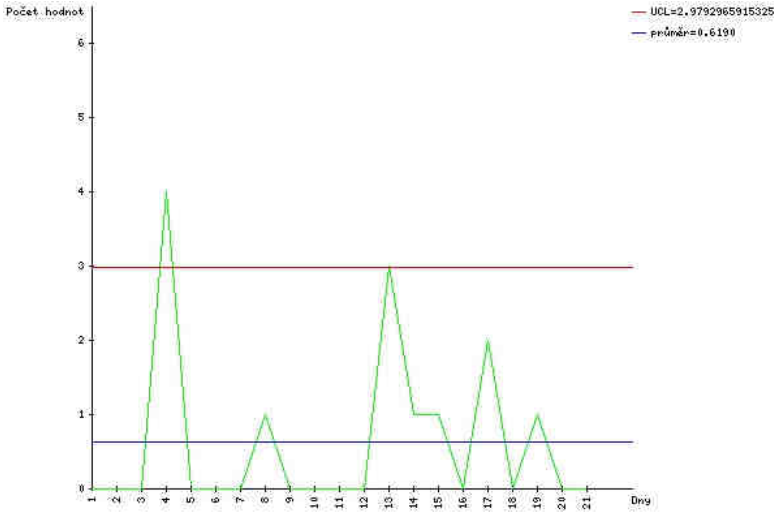
se měří, eventuelně v jakých časových úsecích. Rovnice pro výpočty jednotlivých hodnot pro vytvoření grafu lze najít ve [4] nebo v souboru grafc.php.

Příklad (viz [4]): Počet pádů počítačového systému je zaznamenáván každý den 12 hodin po dobu 21 dní. Naměřené údaje jsou v tabulce tab. 3.

Den	Počet
1	0
2	0
3	0
4	4
5	0
6	0
7	0
8	1
9	0
10	0
11	0
12	0
13	3
14	1
15	1
16	0
17	2
18	0
19	1
20	0
21	0
	0.6190

Tab. 3 – Hodnoty pro c graf

Výsledný kontrolní graf je na obrázku obr. 21.



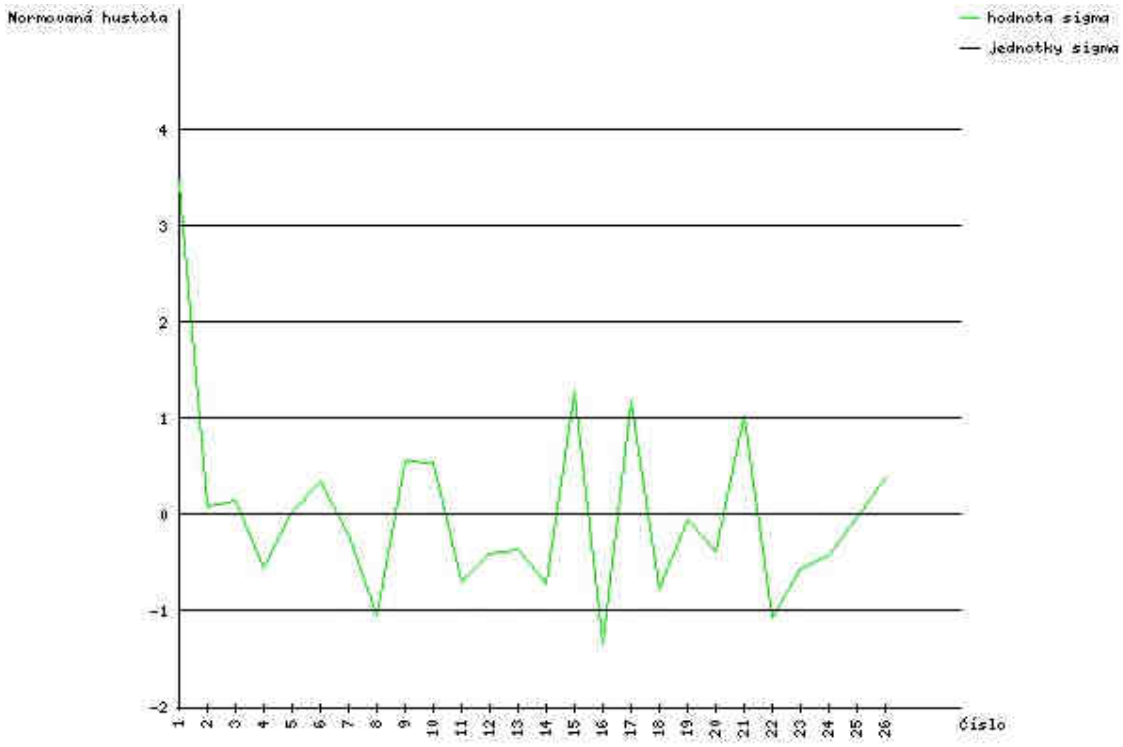
Obr 21 – C graf

Z grafu jsou patrné dvě hodnoty, ve 4. a 13. dni, které způsobují nestabilitu procesu. Pokud se pokusíme tyto události odstranit a stabilizovat tak proces, po přepočítání hodnot se nám ukáže třetí význačný zdroj nestability (17. den). Ten musí být také odstraněn, aby se proces stal stabilním v tomto 21 denním časovém úseku.

5.3.5.3 Měření reprezentovaná z grafem

Toto měření použijeme, chceme-li měřit výskyt nějakých událostí v závislosti na oblasti výskytu, např. na velikosti modulu nebo počtu řádků kódu. Xml šablona pro hodnoty obsahuje číslo měření, počet událostí a velikost charakterizující oblast výskytu. Z těchto zadaných údajů se automaticky spočítá poměr (např . počet chyb na 1000 řádků kódu), dále hodnotu sigma pro jednotlivá měření, $sigma = \sqrt{\frac{pomer0}{velikost}}$, kde pomer0 značí průměrnou hodnotu poměrů ze všech měření, a nakonec spočítáme sigmě odpovídající hodnotu Z pro každý poměr: $Z = \frac{(pomer_i - pomer0)}{sigma_i}$. Tyto hodnoty jsou počítány při ukládání dat z xml souboru ve skriptu parserzgraf.php. Při vkládání hodnocení je samozřejmě opět nutné zadat popis naměřených hodnot.

Příklad (viz [4]): v tabulce tab. 4 jsou zobrazeny výsledky testů 26 softwarových modulů. U každého modulu bylo zadáno jeho číslo, počet nalezených chyb a velikost modulu v počtu řádků kódu. Ostatní hodnoty byly automaticky dopočítány.



Obr. 22 – Z graf

Číslo	Počet	Velikost	Poměr	Sigma	Z
1	19	430	0.04419	0.0069	3.4823
2	8	380	0.02105	0.0073	0.1058
3	3	134	0.02239	0.0123	0.1718
4	6	369	0.01626	0.0074	-0.5419
5	9	436	0.02064	0.0068	0.0532
6	4	165	0.02424	0.0111	0.3575
7	2	112	0.01786	0.0135	-0.1796
8	4	329	0.01216	0.0079	-1.0339
9	12	500	0.02400	0.0064	0.5846
10	8	324	0.02469	0.0079	0.5578
11	6	391	0.01535	0.0072	-0.6842
12	6	346	0.01734	0.0077	-0.3836
13	2	125	0.01600	0.0127	-0.3358
14	8	503	0.01590	0.0063	-0.6894
15	8	250	0.03200	0.0090	1.3017
16	3	312	0.00962	0.0081	-1.3219
17	12	419	0.02864	0.0070	1.2022
18	6	403	0.01489	0.0071	-0.7594
19	3	150	0.02000	0.0116	-0.0238
20	6	344	0.01744	0.0077	-0.3695
21	11	396	0.02778	0.0072	1.0485
22	2	204	0.00980	0.0100	-1.0509
23	8	478	0.01674	0.0065	-0.5430
24	2	132	0.01515	0.0124	-0.4137
25	5	249	0.02008	0.0090	-0.0218
26	10	435	0.02299	0.0068	0.3974

Tab. 4 – Hodnoty pro z graf

Na obrázku obr. 22 je vytvořený z graf z těchto hodnot.

Vidíme, že hodnota v prvním bodě se výrazně liší od ostatních hodnot. Proto je potřeba ji prozkoumat a pokud objevíme zdroj nestability, musíme podniknout kroky k jejímu odstranění. Pokud tento negativní vliv eliminujeme, můžeme znova přepočítat hodnoty k ověření stability procesu. Může se však stát, že první kontrola kódu byla neobyčejně efektivní a potom by bylo potřeba vyřadit ostatních 25 měření a začít znova. Záleží na konkrétních případech a zkušenostech pracovníků měřících proces a zpracovávajících hodnoty.

5.3.6 Modul editace uživatele

K této části systému mají přístup všichni uživatelé. Přidělený pracovník, manažer projektu a manažer procesů zde mohou editovat své osobní údaje kromě loginu, role a údaje o blokování. Ve

formuláři jsou předvyplněny stávající hodnoty kromě hesla, všechny údaje kromě hesla musí být vyplněny. Při změně hesla je nutno zadat nové heslo dvakrát pro vyloučení překlepů.

Uživatel v roli správce systému má právo editovat údaje všech uživatelů. Uživatele může buď vybrat ze seznamu, nebo vyhledat pomocí formuláře. Při editaci má také předvyplněny všechny údaje kromě hesla. Správce systému může také vytvořit nového uživatele, u kterého se automaticky nastaví, že není blokován. Při vytváření nového uživatele se kontroluje zda již v systému není uživatel se stejným jménem, je nutno zadat všechny údaje včetně hesla.

5.3.7 Instalace systému

Systém lze uvést do provozu umístěním na webový server. Musí být zachována adresářová struktura, to znamená celý systém v jedné složce s podadresáři dokumenty, obrazy a procesy. Na serveru musí být nainstalováno PHP a MySQL. Také musí být nastavena přístupová práva na server, zejména do adresářů dokumenty a procesy kvůli přenášení souborů prostřednictvím systému, jinak mohou nastat problémy s přenášenými soubory. Po nahrání systému na server je nutné vytvořit databázi. Na přiloženém CD je soubor instal.txt, obsahující SQL skript pro vytvoření tabulek v databázi a jejich naplnění testovacími daty. Původně jsem vytvořil a testoval tabulky typu InnoDB. Tento typ tabulek podporuje využívání transakcí i používání cizích klíčů pro propojování tabulek a vytváření vztahů mezi nimi. Protože na školním serveru EVA, kde je systém nahrán, je nastaven typ MyISAM, obsahuje soubor install.php SQL skript pro vytvoření tabulek typu MyISAM. Programová část systému je navržena tak aby podporovala oba typy tabulek. Před spuštěním systému je ještě potřeba upravit soubor dbconnect.php doplněním údajů pro připojení k databázi. Na počítači na kterém je program spouštěn musí být pro správnou funkčnost systému povoleny cookies a JavaScript. Vzhled systému je optimalizován pro rozlišení monitoru 800x600 a vyšší.

Systém je možné testovat na adrese
http://www.stud.fit.vutbr.cz/~xdosta15/DP/
login : xadm00
heslo : 123
Uživatel xadm00 je v roli správce systému.

6 Závěr

V této práci jsem se zabýval využitím procesů při vývoji softwaru. Definoval jsem termín proces a popsal možnosti jeho dokumentace. Nastínil jsem možnosti při vytváření projektů s využitím procesů a praktiky používané při zdokonalování procesů.

Také jsem se snažil popsat různé typy organizací podle vyspělosti v procesním řízení. Toto hodnocení může pomoci firmám v jejich rozvoji a pomoci jim zlepšit svoje výsledky a také získat větší důvěru u zákazníků. Podle mého je budoucnost vývoje softwaru právě ve vytváření projektů podle procesů, jenž může společnost vylepšovat a aktualizovat podle nejnovějších poznatků a podle požadavků trhu.

Jedním ze způsobů jak docílit dobrého procesního řízení v organizaci je zavedení a přijetí produktu RUP. Nejlepších výsledků docílíme, pokud si tento způsob vývoje nejdříve vyzkoušíme na nějakém jednodušším pilotním projektu a pak teprve při složitých projektech.

V první části diplomové práce jsem se zabýval problematikou pouze teoreticky, ve druhé části jsem se zaměřil na praktické využití znalostí a vytvořil vlastní aplikaci pro dokumentaci procesů a jejich využití při vývoji sofwarových projektů. V této aplikaci jsem využil poznatků získaných při tvorbě teoretické části a vyšel jsem ze specifikace požadavků uvedené v semestrálním projektu. Tu jsem rozšířil a upřesnil a vytvořil podle ní návrh systému a podle něho poté vytvořil celou programovou podporu diplomové práce. Ta je pomocí HTML, PHP, CSS a JavaScriptu realizována jako webová aplikace spolupracující s databází MySQL.

Programová podpora mj. umožňuje zadávat do systému informace o vytvářených projektech, zahrnovat do projektů jednotlivé procesy, definovat milníky ve vývoji nebo zadávat vstupy a výstupy procesů. Vytváření, mazání a editaci projektů má na starosti uživatel v roli manažera projektů. V databázi je uložena knihovna procesů, obsahující informace o všech procesech. Ty má na starosti uživatel v roli manažera procesů. Může vytvářet nové procesy, editovat stávající nebo je mazat. Také má za úkol sledovat hodnocení jednotlivých procesů a podle něho jednotlivé procesy upravovat a optimalizovat pro použití v dalších projektech. Procesy hodnotí zejména uživatelé v rolích přidělených pracovníků. Tito uživatelé by měli mít určité znalosti, aby byli schopni správně provést měření a navíc při jeho ukládání do systému srozumitelně a věcně uvést všechny potřebné informace potřebné ke zpracování hodnocení.

Systém splňuje všechny požadavky uvedené ve specifikaci a návrhu a je připraven na použití v praxi. Před jeho nasazením by bylo nutné naplnit knihovnu procesů větším množstvím předdefinovaných procesů, jenž by organizace využila při vývoji svých programů. Možným

rozšířením by byla např. implementace vlastního nástroje pro vytváření vývojových diagramů procesů přímo v aplikaci, rozšíření možností hodnocení procesů nebo přidání hodnocení produktů jednotlivých projektů. Také by bylo vhodné rozšířit okruh uživatelů systému možností volby jazykového rozhraní aplikace (zejména angličtiny).

6 Literatura

[1] Becker, J., Kugeler, M., Rosemann, M.: Process Management, Springer-Verlag, 2003, ISBN: 3-540-43499-2

[2] Dyba, T., Dingsoyr, T., Moe, N. B.: Process Improvement in Practice, Kluver Academic Publishers, 2004, ISBN: 1-4020-7869-2

[3] Bergström, S., Raberg, L.: Adopting the Rational Unified Process, Addison-Wesley, 2003, ISBN: 0-321-20294-5

[4] Florac, William A., Measuring The Software Process, 1999, ISBN: 978-0201604443

[5] Kreslíková, J.: Elektronické podklady k přednáškám předmětu RPS, FIT VUT Brno, 15.5.2007, http://www.fit.vutbr.cz/study/course-l.php?id=471

[6] Kreslíková, J., Brejcha, M.: Procesy v moderním managementu, FIT VUT Brno, 2002, ISBN: 80-86596-03-6

[7] domovská stránka programu SmartDraw, 15.5.2007, http://www.smartdraw.com/exp/sof/

[8] článek o RUP, 15.5.2007, http://www.cs.vsb.cz/behalek/frvs/2005/rup/technologie_rup.pdf

[9] problematika ISO_CMMI a ISO 15.5.2007, http://www.dcit.cz/files/jakost/ISO_CMMIxISO.pdf

7 Přílohy

Příloha 1. CD-R

- Prototyp systému pro dokumentaci procesů
- SQL skript install.txt pro vytvoření databáze s testovacími daty
- Obrázkové soubory s diagramy znázorňujícími procesy
- Dokumenty demonstrující vstupy a výstupy procesů
- Šablony .xml souborů pro zadávání hodnocení
- Soubory pro testování ve formátu .xml obsahující hodnocení procesů
- Text diplomové práce ve formátech .doc a .pdf

Adresářová struktura CD

[/]	- install.txt
	- readme.txt
[/data]	- SQL skript install.txt
	- šablony .xml souborů
	- soubory pro testování
[/system]	- prototyp systému
[/system/procesy]	- soubory s diagramy
[/system/dokumenty]	- dokumenty demonstrující vstupy a výstupy procesů
[/dokumentace]	- text diplomové práce